

**IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE**

----- X
NICE SYSTEMS, INC., a Delaware Corporation, and :
NICE SYSTEMS LTD., an Israeli Corporation :
:
Plaintiffs, :
v. : Civil Action No. _____
WITNESS SYSTEMS, INC, a Delaware Corporation, :
Defendant. : **JURY TRIAL DEMANDED**
----- X

COMPLAINT

NICE Systems, Inc. ("NICE") and NICE Systems Ltd. ("NICE, Ltd."), by their attorneys, for their complaint against Witness Systems, Inc. ("Witness") allege as follows:

NATURE OF THIS ACTION

1. This is an action for patent infringement arising under the patent laws of the United States, 35 U.S.C. § 101 et seq., including 35 U.S.C. §§ 154, 271 and 281-285.

PARTIES

2. NICE is a corporation organized and existing under the laws of the State of Delaware with its principal place of business at 301 Route 17, Rutherford, New Jersey, 07070. NICE provides products and services to its customers throughout the United States for the capture, recording, quality monitoring, and analysis of customer interactions for a wide range of markets, including the healthcare, utilities, transportation, retail, financial and telecommunication markets, as well as for outsourcing contact centers.

3. NICE, Ltd. is an Israeli corporation with a principal place of business at 8 Hapnina Street, PO Box 690, 43107 Ra'anana, Israel. NICE, Ltd. is the parent company of NICE.

4. Upon information and belief, Witness is a corporation organized and existing under the laws of the State of Delaware with its principal place of business at 300 Colonial Center Parkway, Suite 600, Roswell, Georgia 30076. Witness sells and offers to sale products and services similar to those provided by NICE throughout the world. Upon information and belief, in April 2003, Witness acquired Eyretel plc, a United Kingdom company located in Leatherhead, United Kingdom, and its domestic subsidiary, Eyretel Inc. located in Calverton, Maryland (hereinafter "Eyretel"). Upon information and belief, upon the combination of Eyretel and Witness, Witness assumed the liabilities of Eyretel for patent infringement occurring before the combination.

JURISDICTION AND VENUE

5. This Court has subject matter jurisdiction pursuant to the provisions of 28 U.S.C. §§ 1331 and 1338(a).

6. This Court has personal jurisdiction over Witness because, *inter alia*, Witness conducts business in this judicial district. Venue is proper in this judicial district pursuant to 28 U.S.C. §§ 1391 and 1400.

COUNT I PATENT INFRINGEMENT OF THE '738 PATENT

7. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

8. NICE is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 5,274,738 ("the '738 patent"), entitled "Modular Digital Voice

Processing.” The ’738 patent was duly and legally issued on December 28, 1993 and is currently valid and enforceable. NICE has the right to sue and recover for past, present and future infringement of the ’738 patent and to obtain the relief sought herein. A copy of the ’738 patent is attached as Exhibit A.

9. The ’738 patent discloses and claims a digital modular voice processing system.

10. Upon information and belief, Witness has been and is infringing the ’738 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, Eyretel’s ContactStore, Eyretel’s MediaStore, Eyretel’s Contact 7000, eQuality ContactStore and ContactStore, that are covered by the claims of the ’738 patent.

11. Witness’ acts alleged herein constitute infringement of the ’738 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

12. Due to Witness’ acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

13. On information and belief, Witness’ infringing activities are deliberate and willful.

14. NICE has no adequate remedy at law.

COUNT II
PATENT INFRINGEMENT OF THE ’371 PATENT

15. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

16. NICE is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 5,396,371 ("the '371 patent"), entitled "Endless Loop Voice Data Storage and Retrievable Apparatus and Method Thereof." The '371 patent was duly and legally issued on March 7, 1995, and is currently valid and enforceable. NICE has the right to sue and recover for past, present and future infringement of the '371 patent and to obtain the relief sought herein. A copy of the '371 patent is attached as Exhibit B.

17. The '371 patent discloses and claims methods, systems and a device for processing, storing and retrieving audio in connection with audio loggers.

18. Upon information and belief, Witness has been and is infringing the '371 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, Eyretel's ContactStore, Eyretel's MediaStore, Eyretel's Contact 7000, eQuality ContactStore, ContactStore, Witness ContactStore for Communication Manager, Witness Quality for Communication Manager and Impact 360, that are covered by the claims of the '371 patent.

19. Witness' acts alleged herein constitute infringement of the '371 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

20. Due to Witness' acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

21. On information and belief, Witness' infringing activities are deliberate and willful.

22. NICE has no adequate remedy at law.

COUNT III
PATENT INFRINGEMENT OF THE '005 PATENT

23. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

24. NICE is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 5,819,005 ("the '005 patent"), entitled "Modular Digital Recording Logger." The '005 patent was duly and legally issued on October 6, 1998 and is currently valid and enforceable. NICE has the right to sue and recover for past, present and future infringement of the '005 patent and to obtain the relief sought herein. A copy of the '005 patent is attached as Exhibit C.

25. The '005 patent discloses and claims, among other things, modular digital recording loggers.

26. Upon information and belief, Witness has been and is infringing the '005 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, Eyretel's ContactStore, Eyretel's MediaStore, Eyretel's Contact 7000, eQuality ContactStore and ContactStore, that are covered by the claims of the '005 patent.

27. Witness' acts alleged herein constitute infringement of the '005 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

28. Due to Witness' acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

29. On information and belief, Witness' infringing activities are deliberate and willful.

30. NICE has no adequate remedy at law.

**COUNT IV
PATENT INFRINGEMENT OF THE '570 PATENT**

31. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

32. NICE is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 6,249,570 ("the '570 patent"), entitled "System and Method For Recording and Storing Telephone Call Information." The '570 patent was duly and legally issued on June 19, 2001 and is currently valid and enforceable. NICE has the right to sue and recover for past, present and future infringement of the '570 patent and to obtain the relief sought herein. A copy of the '570 patent is attached as Exhibit D.

33. The '570 patent discloses and claims methods and systems for recording information regarding telephone calls.

34. Upon information and belief, Witness has been and is infringing the '570 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, Eyretel's Contact 7000, eQuality Balance, eQuality ContactStore for IP, eQuality ContactStore, Witness Quality for Communication Manager, Witness ContactStore for Communication Manager and Impact 360, that are covered by the claims of the '570 patent.

35. Witness' acts alleged herein constitute infringement of the '570 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

36. Due to Witness' acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

37. On information and belief, Witness' infringing activities are deliberate and willful.

38. NICE has no adequate remedy at law.

COUNT V
PATENT INFRINGEMENT OF THE '345 PATENT

39. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

40. NICE is presently the owner by assignment of all right, title and interest in and U.S. Patent No. 6,728,345 (the '345 patent), entitled "System and Method for Recording and Storing Telephone Call Information." The '345 patent was duly and legally issued on April 27, 2004 and is currently valid and enforceable. The patent application for the '345 patent was published on November 15, 2001. NICE has the right to sue and recover for past, present and future infringement of the '345 patent and to obtain the relief sought herein. A copy of the '345 patent is attached as Exhibit E.

41. The '345 patent discloses and claims systems and methods for recording information regarding telephone calls.

42. Upon information and belief, Witness had actual notice of the published patent application for the '345 patent.

43. Upon information and belief, Witness has been and is infringing the '345 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, eQuality Balance, eQuality ContactStore for IP, eQuality ContactStore, Witness Quality for Communication Manager, Witness ContactStore for Communication Manager and Impact 360, that are covered by the claims of the '345 patent.

44. Witness' acts alleged herein constitute infringement of the '345 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

45. Due to Witness' acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

46. On information and belief, Witness' infringing activities are deliberate and willful.

47. NICE has no adequate remedy at law.

**COUNT VI
PATENT INFRINGEMENT OF THE '372 PATENT**

48. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

49. NICE is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 6,775,372 ("the '372 patent"), entitled "System and Method for Multi-Stage Data Logging." The '372 patent was duly and legally issued on August 10, 2004 and is currently valid and enforceable. NICE has the right to sue and recover for past, present and future infringement of the '372 patent and to obtain the relief sought herein. A copy of the '372 patent is attached as Exhibit F.

50. The '372 patent discloses and claims multi-stage data logging systems as well as methods related to, and a data logger used in, such systems.

51. Upon information and belief, Witness has been and is infringing the '372 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, Impact 360, Witness ContactStore, Witness Quality

for Communication Manager and Witness ContactStore for Communication Manager that are covered by the claims of the '372 patent.

52. Witness' acts alleged herein constitute infringement of the '372 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

53. Due to Witness' acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

54. On information and belief, Witness' infringing activities are deliberate and willful.

55. NICE has no adequate remedy at law.

COUNT VII
PATENT INFRINGEMENT OF THE '370 PATENT

56. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set herein.

57. NICE is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 6,785,370 ("the '370 patent"), entitled "System and Method for Integrating Call Record Information." The '370 patent was duly and legally issued on August 31, 2004 and is currently valid and enforceable. The patent application for the '370 patent was published on December 27, 2001. NICE has the right to sue and recover for past, present and future infringement of the '370 patent and to obtain the relief sought herein. A copy of the '370 patent is attached as Exhibit G.

58. The '370 patent discloses and claims methods, computer programs and articles of manufacturing for constructing and maintaining data representations of lifetimes of telephone calls.

59. Upon information and belief, Witness had actual notice of the published patent application for the '370 patent.

60. Upon information and belief, Witness has been and is infringing the '370 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, eQuality Balance, eQuality ContactStore for IP, eQuality ContactStore, Witness Quality for Communication Manager, Witness ContactStore for Communication Manager and Impact 360, that are covered by the claims of the '370 patent.

61. Witness' acts alleged herein constitute infringement of the '370 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

62. Due to Witness' acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

63. On information and belief, Witness' infringing activities are deliberate and willful.

64. NICE has no adequate remedy at law.

**COUNT VIII
PATENT INFRINGEMENT OF THE '920 PATENT**

65. NICE repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

66. NICE is presently the owner by assignment of all right, title and interest in and U.S. Patent No. 6,870,920 ("the '920 patent"), entitled "System and Method for Multi-Stage Data Logging" The '920 patent was duly and legally issued on March 22, 2005 and is currently valid and enforceable. The patent application for the '920 patent

was published on November 07, 2002. NICE has the right to sue and recover for past, present and future infringement of the '920 patent and to obtain the relief sought herein.

A copy of the '920 patent is attached as Exhibit H.

67. The '920 patent discloses and claims methods for accessing information in a digital loggers and related systems.

68. Upon information and belief, Witness had actual notice of the published patent application for the '920 patent.

69. Upon information and belief, Witness has been and is infringing the '920 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, Impact 360, Witness ContactStore, Witness Quality for Communication Manager and Witness ContactStore for Communication Manager, that are covered by the claims of the '920 patent.

70. Witness' acts alleged herein constitute infringement of the '920 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

71. Due to Witness' acts alleged herein, NICE has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE will continue.

72. On information and belief, Witness' infringing activities are deliberate and willful.

73. NICE has no adequate remedy at law.

**COUNT IX
PATENT INFRINGEMENT OF THE '079 PATENT**

74. NICE, Ltd. repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

75. NICE, Ltd. is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 6,959,079 (“the ’079 patent”), entitled “Telephone Call Monitoring System.” The ’079 patent was duly and legally issued on October 25, 2005 and is currently valid and enforceable. The patent application for the ’079 patent was published on August 7, 2003. NICE, Ltd. has the right to sue and recover for past, present and future infringement of the ’079 patent and to obtain the relief sought herein. A copy of the ’079 patent is attached as Exhibit I.

76. The ’079 patent discloses and claims systems for monitoring, based on a predefined condition, the telephonic interactions, including both monitoring and recording screen and audio data, between an agent and a customer.

77. Upon information and belief, Witness had actual notice of the published patent application for the ’079 patent.

78. Upon information and belief, Witness has been and is infringing the ’079 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to, eQuality ContactStore for IP, eQuality ContactStore, Witness Quality for Communication Manager, and Impact 360, that are covered by the claims of the ’079 patent.

79. Witness’ acts alleged herein constitute infringement of the ’079 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

80. Due to Witness’ acts alleged herein, NICE, Ltd. has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE, Ltd. will continue.

81. On information and belief, Witness' infringing activities are deliberate and willful.

82. NICE, Ltd. has no adequate remedy at law.

COUNT X
PATENT INFRINGEMENT OF THE '109 PATENT

83. NICE, Ltd. repeats and realleges each of the allegations contained in paragraphs 1 through 6 as though fully set forth herein.

84. NICE, Ltd. is presently the owner by assignment of all right, title and interest in and to U.S. Patent No. 7,010,109 ("the '109 patent"), entitled "Digital Recording of IP Based Distributed Switching Platform." The '109 patent was duly and legally issued on March 7, 2006 and is currently valid and enforceable. The patent application for the '109 patent was published on June 9, 2005. NICE, Ltd. has the right to sue and recover for past, present and future infringement of the '109 patent and to obtain the relief sought herein. A copy of the '109 patent is attached as Exhibit J.

85. The '109 patent discloses and claims systems for recording and monitoring IP multimedia sessions.

86. Upon information and belief, Witness had actual notice of the published patent application of the '109 patent.

87. Upon information and belief, Witness has been and is infringing the '109 patent within the United States by manufacturing, using, selling and offering for sale products including but not limited to Witness ContactStore, Witness Quality for Communication Manager, Witness ContactStore for Communication Manager and Impact 360 that are covered by the claims of the '109 patent.

88. Witness' acts alleged herein constitute infringement of the '109 patent in violation of the patent law of the United States, 35 U.S.C. §§ 271 and 281-285.

89. Due to Witness' acts alleged herein, NICE, Ltd. has suffered, is suffering, and will continue to suffer irreparable damage, and unless Witness is restrained from continuing its wrongful acts, the damage to NICE, Ltd. will continue.

90. On information and belief, Witness' infringing activities are deliberate and willful.

91. NICE, Ltd. has no adequate remedy at law.

PRAYER FOR RELIEF

WHEREFORE, Plaintiffs demand judgment seeking an Order:

A. Holding that Witness has infringed (by direct, inducement and/or contributory infringement) U.S. Patent Nos. 5,274,738; 5,396,371; 5,819,005; 6,249,570; 6,728,345; 6,775,372; 6,785,370; 6,870,920; 6,959,079 and 7,010,109.

B. Permanently enjoining Witness and its agents, attorneys, servants, successors, assigns, employees, and all those in privy or in active concert and participation with them, or any of them, from infringing U.S. Patent Nos. 5,274,738; 5,396,371; 5,819,005; 6,249,570; 6,728,345; 6,775,372; 6,785,370; 6,870,920; 6,959,079 and 7,010,109.

C. Holding that Witness' conduct was willful and awarding Plaintiffs their reasonable attorney's fees and expenses against Witness pursuant to 35 U.S.C. § 284.

D. Awarding damages to Plaintiffs as a result of Witness' infringement.

E. Awarding a reasonable royalty, pursuant to 35 U.S.C. § 154(d), for U.S. Patent Nos. 6,728,345; 6,785,370; 6,870,920; 6,959,079 and 7,010,109, for the period of

time from the date of publication of each respective patent application through the issuance of the respective patent.

F. Awarding Plaintiffs their attorneys' fees incurred in pursuing this action pursuant to 35 U.S.C. § 285.

G. Awarding Plaintiffs other costs and/or expenses and such other relief as this Court may determine to be just and proper.

JURY TRIAL DEMAND

Pursuant to Fed. R. Civ. P. 38(b), Plaintiffs hereby demand a trial by jury of all issues so triable.


Dated: May 10, 2006

Respectfully submitted,

OF COUNSEL:

KAYE SCHOLER LLP
Scott G. Lindvall
Daniel P. DiNapoli
Joseph M. Drayton

425 Park Avenue
New York, New York 10022
(212) 836-8000

By: 
Josy W. Ingersoll (#1088)
Karen L. Pascale (#2903)
YOUNG-CONAWAY STARGATT
& TAYLOR, LLP
The Brandywine Building
1000 West Street, 17th Floor
Wilmington, DE 19899-0391
(302) 571-6672

*Attorneys for NICE Systems, Inc. and
NICE Systems, Ltd.*

EXHIBIT A



US005274738A

United States Patent [19]

Daly et al.

[11] **Patent Number:** 5,274,738[45] **Date of Patent:** Dec. 28, 1993[54] **MODULAR DIGITAL VOICE PROCESSING SYSTEM**

[75] **Inventors:** Daniel F. Daly, Monroe; John J. Dwyer, Stratford; Mark N. Harris, New Haven; Salvatore J. Morlando, Easton; Thomas C. Grandy, Huntington; Mark Sekas, Orange; Shamla V. Sharma; Jy-Hong Su, both of Norwalk, all of Conn.

[73] **Assignee:** Dictaphone Corporation, Stratford, Conn.

[21] **Appl. No.:** 815,202

[22] **Filed:** Dec. 31, 1991

[51] **Int. Cl.⁵** G10L 9/00

[52] **U.S. Cl.** 395/2

[58] **Field of Search** 381/41-53; 395/2, 275; 379/372, 88, 89, 90, 67; 375/46

[56] **References Cited****U.S. PATENT DOCUMENTS**

4,785,408 11/1988 Britton et al. 395/2
 4,799,144 1/1989 Parruck et al. 379/284
 4,959,855 9/1990 Daudelin 379/213
 4,975,941 12/1990 Morganstein et al. 379/213
 4,991,217 2/1991 Garrett et al. 381/43

Primary Examiner—Michael R. Fleming

Assistant Examiner—Michelle Doerfler

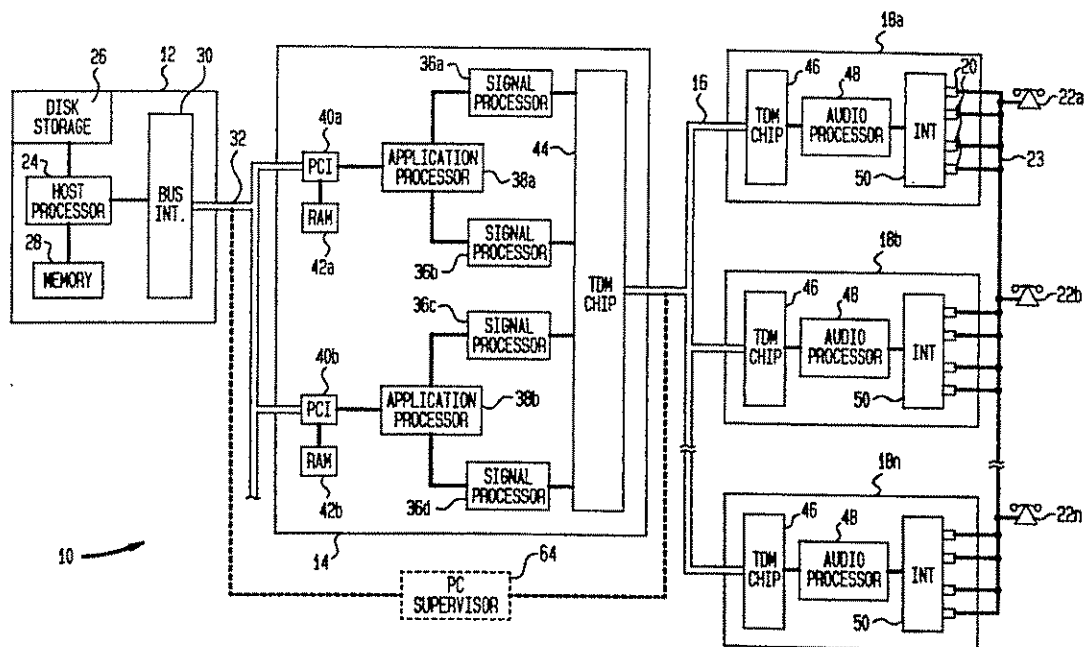
Attorney, Agent, or Firm—Peter Vrahotes; Melvin J. Scolnick

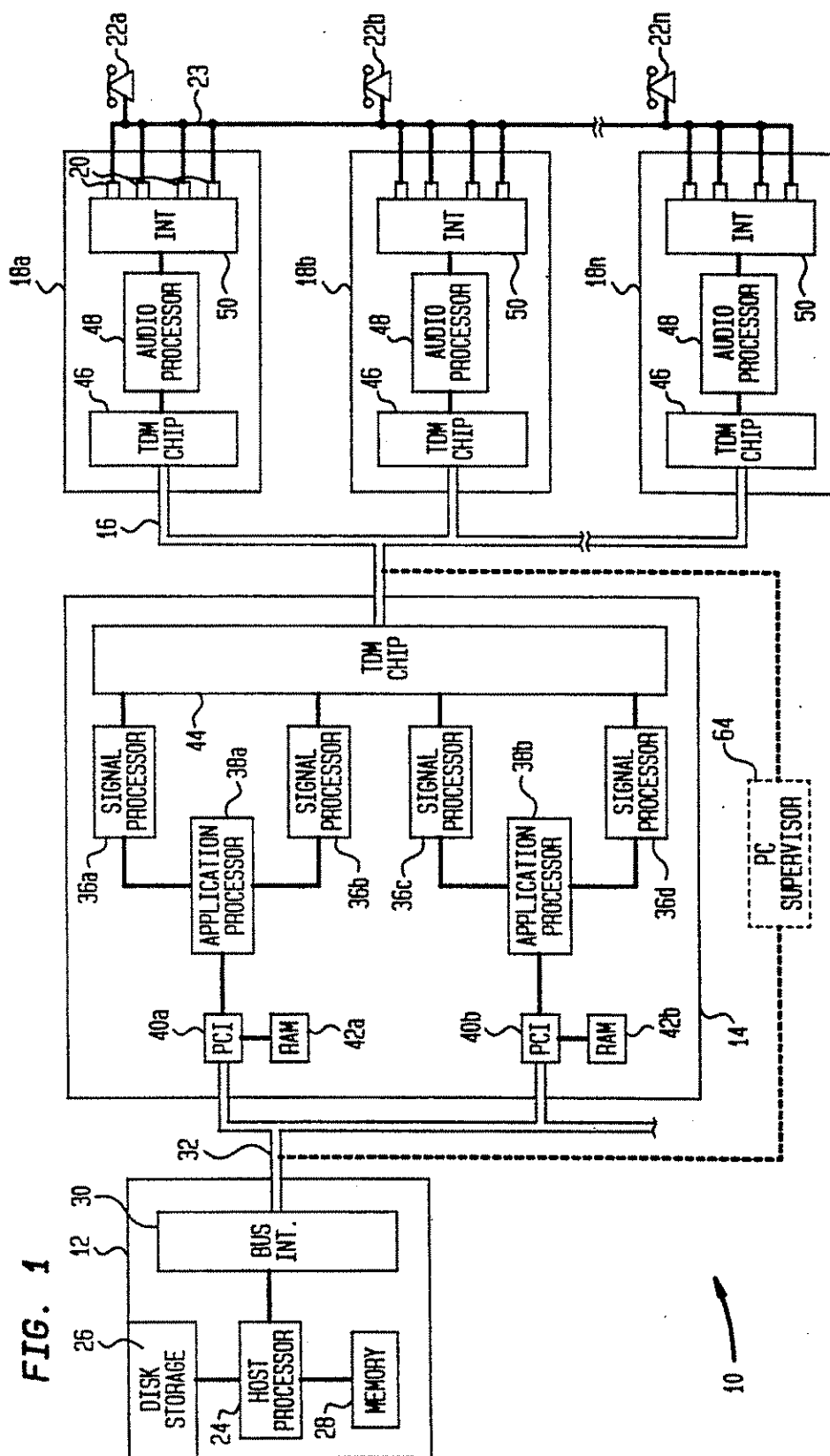
[57]

ABSTRACT

A digital voice processing system wherein voice processing functions are run in software. This application of software allows a modular structure because the application software resides in boards that are coupled to a host computer. With this structure, the software can be updated as required and the capacity of the system can be readily expanded to meet increased needs.

10 Claims, 2 Drawing Sheets



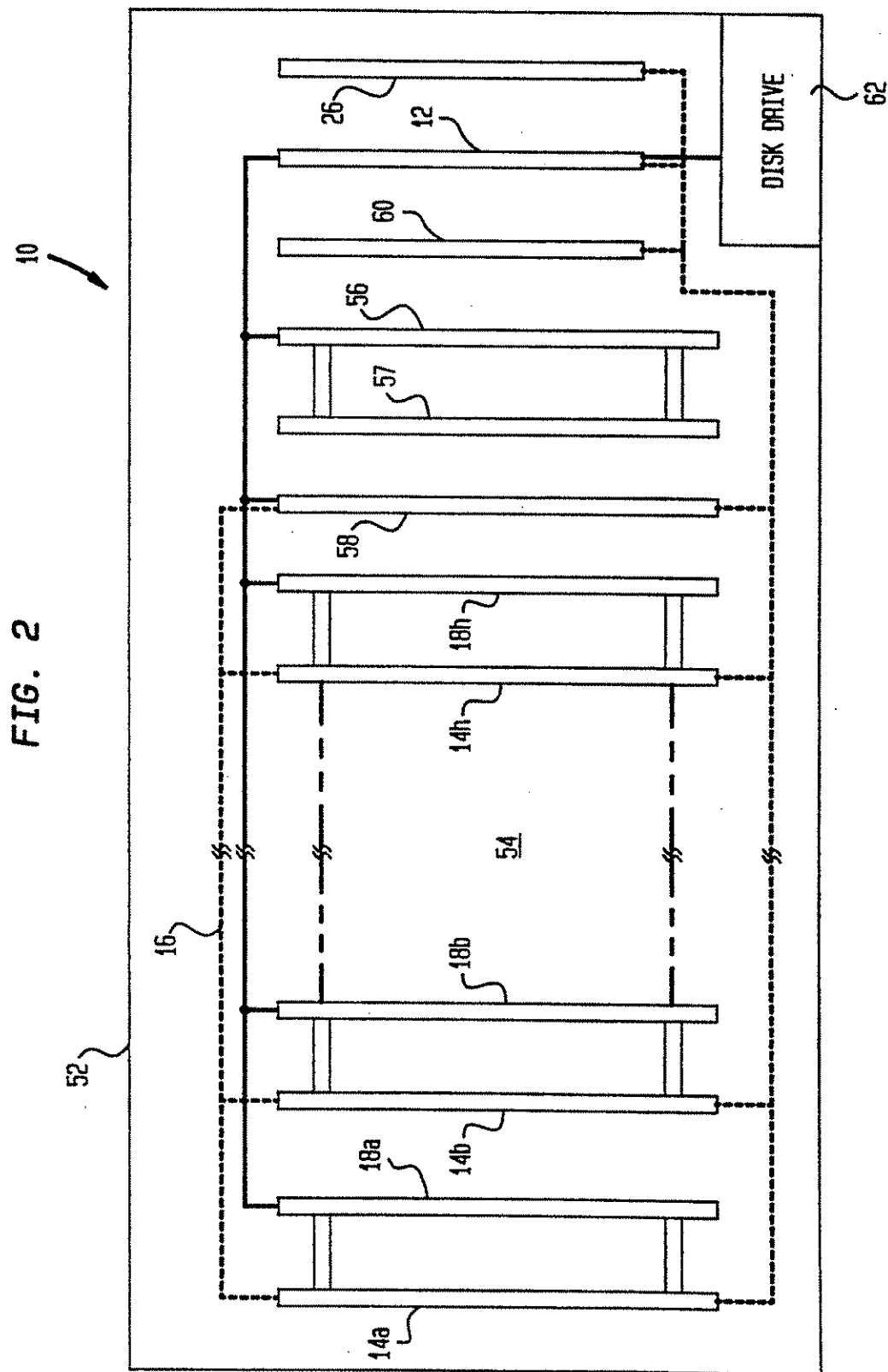


U.S. Patent

Dec. 28, 1993

Sheet 2 of 2

5,274,738



1

5,274,738

2

MODULAR DIGITAL VOICE PROCESSING SYSTEM

BACKGROUND OF THE INVENTION

Digital voice processing systems are known that communicate with telephone systems to perform functions such as voice signal compression, storage and retrieval, automatic gain control, voice activated operation, telephone functions and the like. These functions are performed by such systems in hardware which was costly and inflexible. In addition, expansion of a prior voice processing system was difficult because of the need for additional hardware, not only because of the expense associated therewith, but also because of the geography factor, i.e., a larger footprint was required.

With the ever increasing change in technology, particularly software, it would be advantageous to be able to provide a software based digital voice processing system that is capable of being quickly, conveniently and inexpensively expanded. In addition, it would be advantageous to provide a digital voice processing system which is capable of handling a large amount of data and which is capable of being quickly conveniently and inexpensively expanded as required.

SUMMARY OF THE INVENTION

A modular digital voice processing system has been conceived and developed wherein voice processing functions are run in software. This allows a modular structure whereby units can be readily added or removed. The application software resides in circuit boards that are coupled to a host computer so that the software can be changed as applications require and the number of boards can be increased readily for greater capacity. The host computer is in communication with one or more voice processing circuit boards that perform digital voice processing, and telephone signal processing and application processing. The voice processing circuit boards are in communication with one or more audio circuit boards that digitize data received from outside devices. The voice processing circuit boards communicate with the audio circuit boards through a time division multiplexer bus (TDM). Each audio circuit board includes an analogue unit that receives analogue signals from direct connect and loop start telephones, PBX's and the like, converts the analogue signals to digital and sends the digital data to a signal processor that is used as a high speed multiplexer. The signal processor then sends the signals through a TDM chip onto the TDM bus and they are subsequently received by the voice processing circuit board. The voice processing circuit board performs signal compression, automatic gain control, voice activated operation and application processing. Subsequent to the processing taking place in the voice processing board, data is forwarded from the voice processing circuit board to the host computer for further processing and storage.

BRIEF DESCRIPTION OF THE DRAWING

FIG. 1 is a block diagram of a system in which the instant invention can be practiced, and

FIG. 2 is a plan view of the system shown in FIG. 1.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT(S)

With reference to FIG. 1, a digital voice processing system is shown general at 10 that has a host computer

12, a voice processing circuit board, referred to hereinafter as voice processing card 14, and a time division multiplexer (TDM) bus 16 that connects the voice processing card to a plurality of audio circuit boards 18a, 18b . . . 18n, each of which hereinafter will be referred to as an audio card. Each audio card 18a, 18b . . . 18n has four ports 20 through which communication can be had with a plurality of devices such as direct connect and loop start telephones 22a, 22b . . . 22n, PBX's and the like through telephone lines 23. The telephones can be used to issue commands to the system 10 through DTMF tones. One of the features of the system 10 is that it can act as a telephone switch system.

The host computer 12 can be any of a number of commercially available computers such as an IEEE 996 Standard PC/AT which includes a processor 24, which is in communication with a disk storage 26 and a memory 28. The processor 24 is also in communication with a bus interface 30. The disk storage 26 acts as a storage medium for storing prompts, operating data base directory information and other data. It also serves as back-up memory when the capacity of the memories in the voice processing card 14 are exceeded. Prompts are recorded messages, instructions and menus that are for the purpose of assisting a caller in the use of the voice processing system 10. The memory 28 is a volatile memory which receives the overating code for the system 10 from the disk storage 26 on start-up. The memory 28 also stores diagnostic information and serves as a buffer. The bus interface 30 provides communication between the processor 24 and the voice processing card 14 through a bus 32.

The voice processing card 14 has essentially two independent circuits therein which will be described simultaneously. The voice processing card 14 is shown and described in greater detail in concurrently filed patent application having Ser. No. 07/815,207 and entitled Digital Signal Processing Circuit Board which is hereby incorporated by reference. Each circuit has a host computer interface (PCI) chip 40a, 40b to which a RAM 42a, 42b, respectively, is connected for temporary storage of data and storage of the operating code received from the host computer 12 during initialization. Details of the PCI chip 40a, 40b are given in concurrently filed patent application Ser. No. 07/816,516 and entitled Interface Chip for a Voice Processing System, which is hereby incorporated by reference. Each interface 40a, 40b is in communication with an application processor 38a, 38b, respectively, such as an Intel SOC186. The application processors 38a, 38b run the application processing and database management. Each application processor 38a, 38b is in communication with and controls a pair of signal processors 36a and 36b and 36c and 36d, respectively, which may be a TMS 320C25 processor from Texas Instruments.

All the signal processors 36a-36d are in communication with a time division multiplexer (TDM) chip 44 which is in communication with the bus 16. Details of the TDM chip are shown and described in concurrently filed patent application having Ser. No. 07/441,491 and entitled Time Division Multiplexer Chip and Process Thereof which is hereby incorporated by reference. The signal processors 36a-36d perform voice compression and expansion, depending upon the direction of the data stream, tone detection, voice activated operation, VOX, voice operated recording, automatic gain con-

3

5,274,738

4

trol, control information decoding and telephone call processing.

Each audio card 18 is in communication with the TDM chip 44 through the bus 16 and includes a time division multiplexer (TDM) chip 46 which is identical to the TDM chip 44 of the voice processor card 14 except that it has fewer components connected since it only communicates with one audio processor 48. Details of the audio card 18 can be found in concurrently filed patent application Ser. No. 07/815,205 and entitled Audio Circuit Board for a Modular Digital Voice Processing System, which is hereby incorporated by reference. The TDM chip 46 is in communication with a high speed, processor 48 such as a TMS 320C10 available from Texas Instruments, the latter being in communication with an analogue interface 50 which interfaces through ports 20 with telephones 22a, 22b . . . 22n, through telephone lines 23. The analogue interface 50 can also communicate through the ports 20 with public switch networks, private branch exchanges (PBX) and the like. Optionally, a PC supervisor 64 can be attached to the host computer 12 through an RS232 link for the purpose of providing a keyboard and a screen through which a supervisor can supervise or monitor the system 10.

With reference to FIG. 2, the lay out the system 10 of the digital voice processing system 10 is shown in plan view. The system 10 includes a housing 52 having a base 54 to which the voice processing boards 14 and audio cards 18 are physically attached in pairs without necessarily being logically connected so that the cards can be logically intermixed with one another. More specifically, and by way of example, the voice processing card 14b can be physically connected to audio card 18b but logically connected to audio card 18a. In FIG. 2 the system 10 is shown having eight pairs of voice processing cards 14a-14b and audio cards 18a-18b but some of the voice processing cards can be replaced with dummy cards that only provide physical support and electrical connections to the audio cards without logic. Also included is a sixteen port audio card 56, a clock buffer 58, a local area network (LAN) card 60, the host computer 12, a disk drive 62 and the disk storage 26. A bus 44 connects the host computer 12 to the audio cards 18, 56 and clock buffer 58 so as to control access of the audio processor to locations of RAMs within the TDM chip 46. The host computer 12 can be attached optionally to a PC supervisor 64, see FIG. 1, that would provide a keyboard and monitor that would be beneficial when the system 10 is expanded and would also allow a supervisor to communicate with any memory location in the system 10 and duplicate the same on a different location so that conference calls could take place. Other functions could be performed through the PC supervisor such as diagnostics. The voice processing cards 14 have the capacity to serve a number of audio cards 18 and also serve a 16 port audio card.

By way of example one operation will be described, but it will be appreciated that the system 10 is capable of a variety of functions as indicated in the disclosure. A phone message will be transmitted from a telephone 22, and a signal will be received by the analogue interface 50 of one of the audio cards 18a, 18b-18n. Although only the operations of one telephone 22 and one audio card 22 will be described, it will be appreciated that any one of the telephones 22a, 22b . . . 22n or and any one of the audio cards 18a, 18b . . . 18n could be involved and would function in the same manner. The analogue inter-

face 50 is a loop start type of interface and handles all telephone communications on a first come, first call basis. The analogue interface 50 transforms incoming analogue signals to digital signals, balances the impedance of the telephone 20 system to that of the system 10, and optimizes signal integrity. A digital signal will be sent to the audio processor 48 which is a fast acting signal processing chip. A signal will be sent through the TDM chip 46, onto the TDM bus 16 and will be subsequently received by the TDM chip 44 and a signal processor 36. The signal processor 36a will receive the signal that a telephone 22 is requesting service from an audio process 48. The signal will be sent to an application processor 38a. The response to the request for service will be controlled by the application processor 38a which will direct the signal processor 36a to signal the audio processor to go off hook.

After the audio processor 48 is told to go off hook, communication is established with the telephone user and prompts would give instructions thereto. The user would then respond to the prompts by sending DMTF signals to the system 10 through use of the telephone keyboard.

Data is then received by a signal processor 36 where processing such as speech compression and expansion, call programming, automatic gain control, dual tone multi-frequency extraction, and voice activated operations takes place. The application processor 38 performs high level application such as dictation, transcription, voice mail, voice response, medical records, and the like. Each application processor 38a, 38b can run any of the different types of application processing and can run two applications of the same type simultaneously.

The data is then forwarded from the application processor 38 over the bus 32 to the bus interface 30 by way of the PCI chip 40a 40b and it is subsequently forwarded to the processor 24 informing it of the transaction so that the processor can control data traffic. Voice data is stored in a RAM 42a until the RAM's capacity is exceeded after which the data will be transmitted to the disk storage 26 for subsequent retrieval. The host processor 24 acts as a manager for voice data going into the disk storage 26 and will control specific operations of the system 10 such as systems diagnostics, voice file management and memory location assignments.

Thus what has been shown and described is a modular digital voice processing system wherein components, software, and applications can be readily changed without the need of replacing hardware.

What is claimed is:

1. A digital modular voice processing system comprising:

- a host computer having a host processor, and a storage medium, a memory and a bus interface in communication with said host processor,
- a first bus in communication with said bus interface,
- a voice processing card having at least one digital signal processor and at least one application processor in communication with said at least one digital signal processor, a first interface providing communication between said at least one application processor and said first bus, and a first time division multiplexer chip in communication with said at least one digital signal processor,
- a second bus in communication with said first time division multiplexer chip, and

5,274,738

5

e) at least one audio card including a second time division multiplexer chip that communicates with said second bus, an audio processor in communication with said second time division multiplexer chip, and a second interface in communication with said audio processor, said second interface having a plurality of ports that provide communication with communication lines.

2. The system of claim 1 wherein said voice processing card includes two application processors, two first interfaces each in communication with an application processor on a one to one basis and two pairs of digital signal processors each pair in communication with one of said application processors on a one to one basis, each of said voice processors being in communication with said first time division multiplexer chip.

3. The system of claim 2 wherein said storage medium is a storage disk.

4. The system of claim 1 wherein said second interface of said audio card has means for converting analogue signals received from said ports to digital signals and converting digital data received data received from said audio processor to analog signals.

5. The system of claim 4 wherein said signal processor has means for performing voice compression and expansion, automatic gain control, dual tone multi-frequency extraction and voice activated operations.

6. The system of claim 5 wherein said application processor includes means for performing dictation, transcription, voice mail, voice response and medical records.

7. The system of claim 1 including a housing, said host computer, said first bus, said at least one voice processing card, said second bus, and said at least one audio card being supported by said housing and said at least one voice processing card and said at least one audio card are physically and electrically connected, and said host computer is logically connected to said

6

voice processing card and to said at least one audio card.

8. A digital modular voice processing system comprising:

a) an interface for receiving analogue voice signals from a telephone line and converting said analogue voice signals to digital voice data,

b) an audio processor in communication with interface,

c) first time division multiplexing means in communication with said audio processor for receiving and multiplexing said digital voice data and acting as temporary storage for data,

d) a bus in communication with said first time division multiplexing means for receiving digital voice data therefrom,

e) second time division multiplexing means in communication with said bus for multiplexing digital voice data received from said bus,

f) voice data processing means in communication with said second time division multiplexing means for receiving multiplexed digital voice data and performing digital processing operations on the multiplexed digital voice data,

g) application processing means in communication with said voice data processing means for performing application processing on the processed digital voice data received from said voice data processing means, and

h) a host computer in communication with said application processing means for receiving and storing application processed digital voice data.

9. The system of claim 8 wherein said host computer is in logical communication with said audio processor.

10. The system of claim 8 wherein said voice data processing means performs voice compression and expansion; automatic gain control, extracts dual tone multi-frequency extraction and voice activated operations.

* * * * *

EXHIBIT B



US005396371A

United States Patent [19]

Henits et al.

[11] **Patent Number:** 5,396,371[45] **Date of Patent:** Mar. 7, 1995

[54] **ENDLESS LOOP VOICE DATA STORAGE AND RETRIEVABLE APPARATUS AND METHOD THEREOF**

[75] **Inventors:** John Henits, Bethel; Robert B. Swick, Stratford; Constantine P. Messologitis, Milford; Christopher S. Goane, Greenwich, all of Conn.

[73] **Assignee:** Dictaphone Corporation, Stratford, Conn.

[21] **Appl. No.:** 171,296

[22] **Filed:** Dec. 21, 1993

[51] **Int. Cl.⁶** G11B 5/00; G11B 5/09

[52] **U.S. Cl.** 360/5; 360/32

[58] **Field of Search** 379/89, 45, 88, 53; 360/46, 51, 5, 32, 33.1

[56] **References Cited****U.S. PATENT DOCUMENTS**

4,663,779	5/1987	Szeto	379/88 X
4,811,376	3/1989	Davis et al.	379/88 X
4,891,835	1/1990	Leung et al.	379/45 X
5,034,975	7/1991	Grimes	379/89 X
5,065,428	11/1992	Mitchell et al.	379/89 X
5,163,085	11/1992	Sweet et al.	379/89
5,179,479	1/1993	Ahn	
5,195,128	3/1993	Kinl	379/89 X
5,199,062	3/1993	Van Meister et al.	379/89 X

5,235,475	8/1993	Tokumatsu et al.	
5,283,818	2/1994	Klausner et al.	379/88 X
5,339,203	8/1994	Henits et al.	360/39

FOREIGN PATENT DOCUMENTS

13260001	3/1970	United Kingdom	360/5
----------	--------	----------------	-------

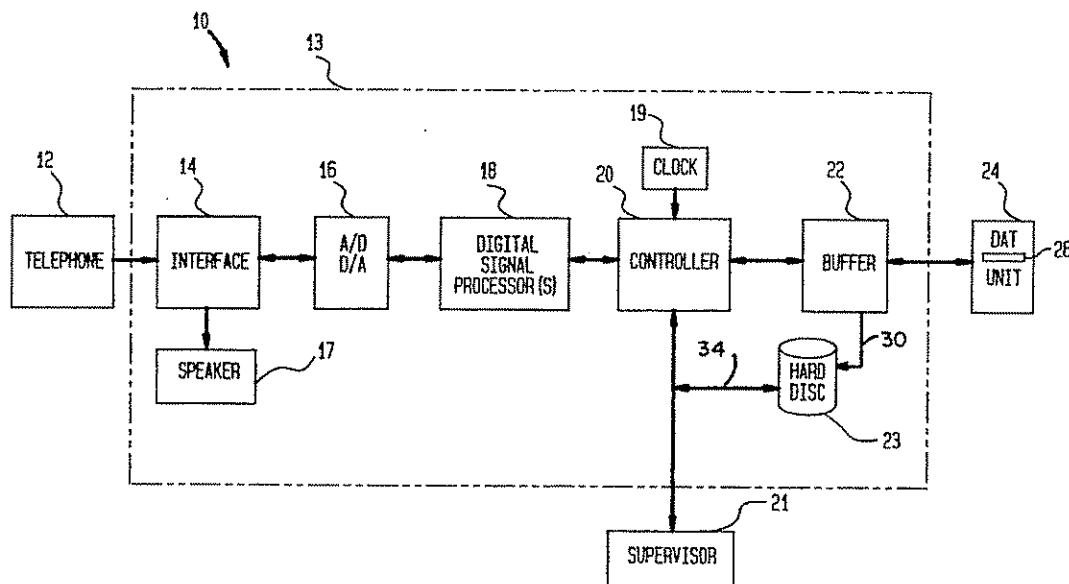
Primary Examiner—Donald Hajec

Assistant Examiner—Thien Minh Le

Attorney, Agent, or Firm—Ronald Reichman; Melvin J. Scolnick

[57] **ABSTRACT**

Apparatus and method for storing and retrieving audio data simultaneously. A digital audio logger is provided with a digital audio tape (DAT) for permanent storage of audio data and with a random access storage (RAS) device that provides fast retrieval of audio. Both the DAT and RAS devices have audio written from a buffer that initially stores audio data temporarily. Two pointers are provided for the RAS device. Because of the randomness characteristic of the RAS device, data can be retrieved rapidly with a first one of the pointers and for this reason has advantages in fast retrieval. The RAS device communicates with a supervisor whereby data can be retrieved therefrom through the first pointer while the RAS device is receiving data from the buffer through the second pointer.

8 Claims, 4 Drawing Sheets

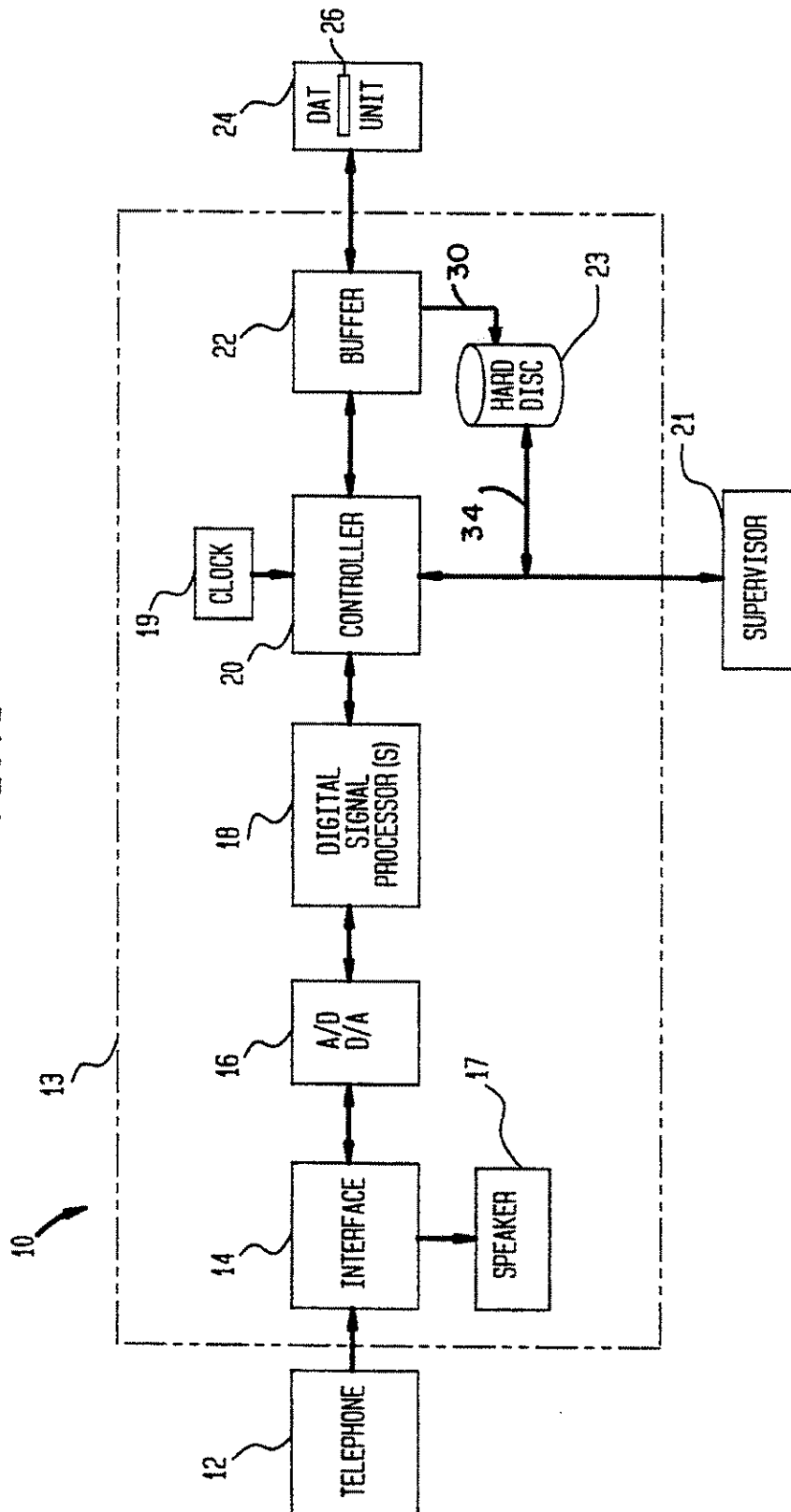
U.S. Patent

Mar. 7, 1995

Sheet 1 of 4

5,396,371

FIG. 1



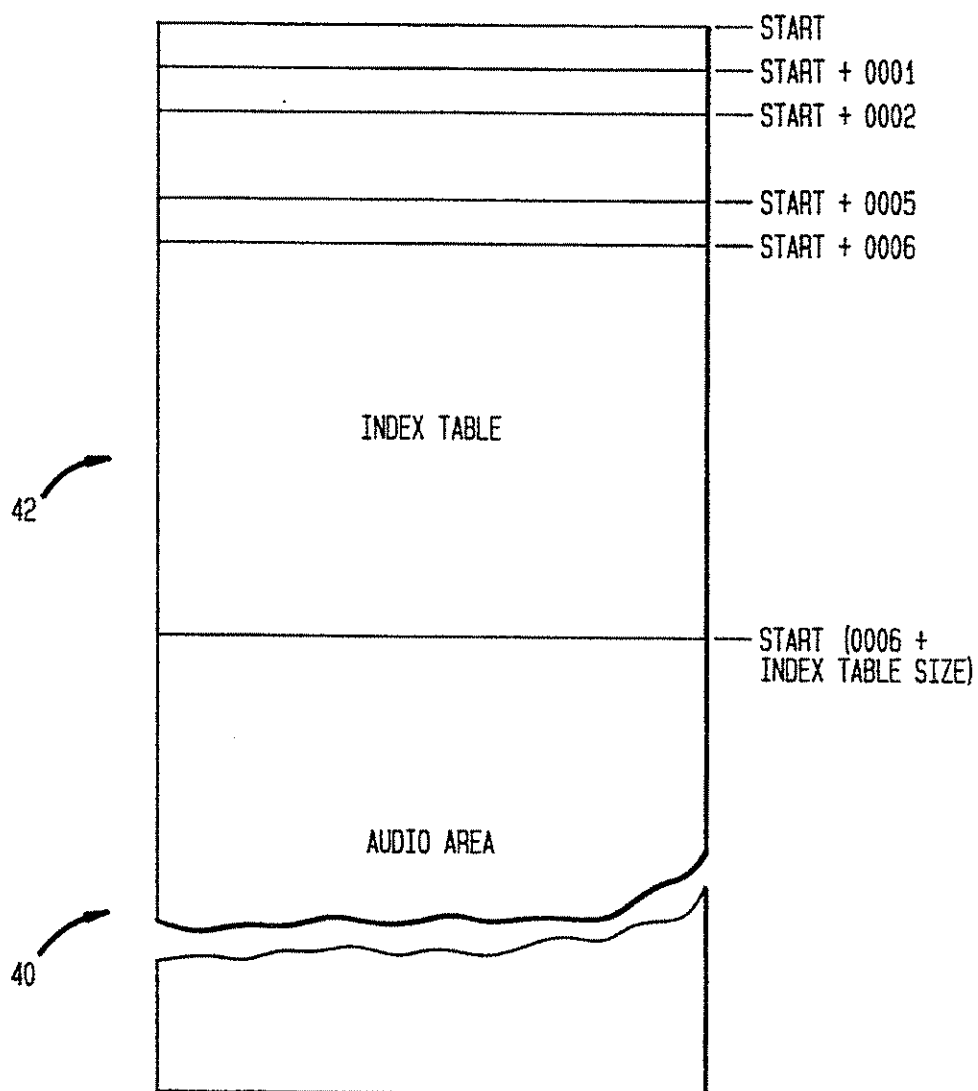
U.S. Patent

Mar. 7, 1995

Sheet 2 of 4

5,396,371

FIG. 2



U.S. Patent

Mar. 7, 1995

Sheet 4 of 4

5,396,371

FIG. 4
SYSTEM TABLE STRUCTURE

AUDIO FRAME FORMAT

HEADER	1 - n BLOCKS
--------	--------------

INDEX TABLE ENTRY FORMAT

SCSI RECORD NUMBER

RST ENTRY FORMAT

ST. TIME	END TIME	INDEX ST. ENTRY	INDEX END ENTRY	CHANNELS
----------	----------	-----------------	-----------------	----------

5,396,371

1

2

ENDLESS LOOP VOICE DATA STORAGE AND RETRIEVABLE APPARATUS AND METHOD THEREOF

BACKGROUND OF THE INVENTION

Audio loggers are well known devices that are used for the purpose of obtaining records of voice communication by recording audio on a tape. They have particular use in police stations, hospitals, prisons, brokerage houses and other locations where there is a need to record a conversation or other audio and the time and date thereof. Upon completion of recording, the tapes upon which audio is written are stored for archival purposes. The tapes can be either digital tapes or analog tapes, depending upon whether a digital audio logger or analog audio logger is used.

One requirement for a logger is that there be an ability to have audio retrieved even though a tape is still having audio written thereon. Clearly, if a tape had to be repositioned in order to obtain audio, there would be an interruption and data would be lost. In order to avoid such an occurrence, prior loggers have provided redundant tapes. One tape would act as a primary tape that receives data on a continuous basis while the second tape receives the same data simultaneously, except that the second tape could be removed when audio had to be retrieved. Upon removal of the secondary tape, another secondary tape would be installed in the logger and the redundancy is continued.

Although the redundancy scheme provides a way for obtaining data without interrupting recording, there clearly are disadvantages. One disadvantage is that an audio tape is a slow responding medium and searching for information on a tape is time consuming. This is a disadvantage during the time of emergencies. Another disadvantage is that a lapse in written data could occur during exchange of the secondary tapes. Clearly, it would be advantageous to be able to have continuous recording of audio without the disadvantages of using redundant tapes.

SUMMARY OF THE INVENTION

Apparatus and method have been devised wherein information can be retrieved from a digital audio logger as the logger continues to receive audio. The audio logger is provided with a buffer that receives audio in real time and temporarily stores the same in the buffer. A digital audio tape (DAT) and a random access storage (RAS) device are in communication with the buffer to simultaneously receive data when the buffer down loads data. This occurs after the buffer has stored a prescribed amount of data.

A supervisor is in communication with the logger, and a data retrieving pointer is connected to the RAS device so that data can be retrieved from the RAS device while data is also being written thereto by a second pointer. The data retrieving pointer first goes to the header of the RAS device for determining the location of data stored at a particular time. After the pointer reads the location of data from the header, it will then contact the location at which the desired information is written. The recording will be played back so that the sought after information can be retrieved. As such retrieving is taking place, the second pointer allows the RAS device to continuously receive data at the same time as the DAT.

BRIEF DESCRIPTION OF THE DRAWING

FIG. 1 is a block diagram illustrating a digital audio logger system in which the instant invention can be practiced;

FIG. 2 is a schematic representation of the random access storage device shown in FIG. 1 illustrating the data sections thereof;

FIG. 3 is a schematic representation of the manner in which data is retrieved from the RAS device illustrated in FIG. 2; and

FIG. 4 is a system table structure that describes the format for the data.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

With reference to FIG. 1, a digital audio logger system is shown generally at 10, in which the instant invention can be practiced. A plurality of audio sources 12, such as telephones, are able to transmit audio to an interface 14 of a digital logger 13 which monitors the telephones. Although the invention will be described with the use of telephones 12, it will be appreciated the invention can be used with other sources of audio, such as police radios. The interface 14 is in communication with a speaker 17 and with an analog/digital (A/D), digital/analog (D/A) converter 16 that will convert analog signals received from the telephones 12 to digital signals when data is flowing in one direction and digital to analog signals when data flows in the opposite direction. A digital signal processor 18 is in communication with the converter 16 and performs the function of compressing the digital voice signals by use of a voice compressing algorithm as is known in the art. Any of a number of commercially commercially available signal processors can be used for the purpose, such as a Texas Instrument TMS 320C25 processor available from Texas Instruments Inc. The compressed data is received by a controller 20 that arranges the data in a prescribed order and controls the flow of the data. In with the controller 20, is a clock 19 that provides the time and data, a buffer 22 that temporarily stores data, a random access storage device 23 and a supervisor 21 that provides access to the logger 13.

The buffer 22 is a memory that communicates with a digital audio tape (DAT) drive 24 and a random access storage (RAS) device 23. The DAT drive 24 is adapted to receive a DAT tape 26. A pointer 30 provides communication from the buffer 22 to the RAS device for the purpose of transferring data to the RAS device that is temporarily stored in the buffer. A second pointer 34 is also in communication with the RAS device 23 and is under control of the controller 20 in response to input from the supervisor 21 as will be explained hereinafter.

With reference now to FIG. 2, the random access storage device 23 is shown in schematic form and is composed of two portions, or partitions, a primary partition 40 and a secondary partition 42. The primary partition 40 is divided into physical locations identified by SCSI record numbers and representing dynamic time slots, with each time slot corresponding to 6 seconds of audio.

The term dynamic time is used to accent the fact that time represented by a time slot changes as audio is written. Data for a given time thereof will first be written into slot 1, and this process will progress until the last slot is filled. Thereafter, the sequence will start from the beginning again with the new data being written into

5,396,371

3

slot 1 to for another given time replace the original data. The pointer 30 initially writes data into the top of the primary partition and continues writing data until the primary partition is full. Thereafter, the pointer 30 will again be located at the top of the primary partition and it will erase prior data while writing incoming data. It will be appreciated that an RAS memory can be of variable capacity so that it can be provided with more memory than the DAT, so as to be able to store more data than a DAT. Thus, an RAS device will store all the data storable in a tape thereby inhibiting the loss of data.

The secondary partition has two tables, a record session table 45 and an index table 46. The record session table 45 keeps track of the time when the logger 13 is recording and the index table 46 stores SCSI record numbers which is the location data in the primary partition 40.

With reference now to FIG. 3, a block diagram representation of the RAS device 23 is shown with the record session table 45, index table 46 and the primary partition 40 shown separately to enhance explanation of the recording and retrieving of data. The record session table 45 contains the start time, and end time columns of a recording session and columns for the location can be found in the index table for the beginning and end of message. In short, the record session table acts as a pointer for the index table 46. The index table 46 acts as a pointer for the primary partition and has a column of implied time slots, each representing 6.0 seconds, and a SCSI record number location column indicating where audio for a particular time slot is located on the primary partition 40. Although the time is shown only in terms of hours and minutes for convenience, it will be appreciated that the date will also be included.

With reference to FIG. 4, the audio is divided into a plurality of frames, each frame including a header and a plurality of audio blocks, as for example 5. Each frame will be allocated a period for recording, such as 6 seconds of time, as opposed to an amount of audio. The index table entry format is made up of a plurality of SCSI record numbers and determines the location of a frame within the primary partition 40 by assigning a SCSI record number to the same. It will be appreciated that even though a frame is allocated six seconds, the time of recording can be less or greater depending on the activity of the telephone and the number of telephones (channels) being monitored.

The RST entry format, as implemented in the record session table is made up of a start time, an end time, an index start entry, an index end entry and the recorded channels for the purpose of furnishing a selected frame or combination of the frames in the primary partition.

In operation, audio will be entered through the telephones 12, be received by the interface 14 and converted from analog to digital by the converter 16. Thereafter, the digital signals are received by the digital signal processor 18 that will compress the signals. These compressed signals will be received by the controller and the time of receipt will be correlated with the clock 19. What has been shown and described heretofore are known components of a digital logger system and will not be described in detail. The controller 20 sends the audio to the buffer 22 and the buffer 22 temporarily stores the audio. Thereafter, the buffer 22 transmits the data to a tape 26 that is housed in the DAT drive unit 24 and also transmits the data to a RAS device 23, the data being received simultaneously by these two units 23, 24.

4

The supervisor 21 can communicate with the controller 20 for the purpose of obtaining data from the RAS device 23. Upon input of the time and date of the data to be retrieved, the controller will locate the pointer 34 first at the appropriate location on the record session table 45 in the secondary partition to determine the location on the index table 46 of the SCSI record number. Upon such SCSI record number being located in the index table 46, the pointer 34 will seek such location to determine the SCSI record number where the sought after audio is found on the primary partition 40. The pointer 34 will go to such location so that the data can be played back through the speaker 17. The request for the retrieving of data has no effect on the pointer 30 which continues to transmit data to the RAS device first by supplying audio on the primary partition, then communicating with the secondary partition to write data for the tables 45, 46. Because of the separation of the primary partition into SCSI records, the voice data can be retrieved within 6 seconds of recording.

With reference now to FIG. 1, as the the first pointer 30 advances along the primary partition of the RAS device 23, it will continue to write data into the primary partition of RAS 23 until it reaches the last available memory slot. Thereafter, it will shift and start again at the beginning of the primary partition, replacing prior data with newly received audio, thus forming an endless loop. As stated previously, the pointer will shift from the primary partition to the secondary and back to refresh the data in the tables 45, 46 and record audio in the primary partition.

Thus what has been shown and described is an apparatus and method whereby audio can be retrieved from a digital audio logger while audio is still being recorded therein. This is accomplished by the use of a random access storage device that has two pointers, a first pointer for receiving data and the second pointer for retrieving data.

The above embodiments have been given by way of illustration only, and other embodiments of the instant invention will be apparent to those skilled in the art from consideration of the detailed description. Accordingly, limitations on the instant invention are to be found only in the claims.

What is claimed is:

1. In a method of storing and retrieving audio from a digital audio logger, the steps comprising:

monitoring an audio source,
storing audio data from the audio source in a buffer,
writing the audio data from the buffer onto a digital audio tape and a random access storage device, and
retrieving audio from the random access storage device while audio data is written into the digital audio tape and the random access storage device.

2. The method of claim 1 including the further steps of providing the random storage device with a primary partition and writing voice data onto the primary partition in time defined manner.

3. The method of claim 2 further including the further steps of providing the random access device with a secondary partition and writing an index table in the secondary partition to indicate location of audio data in the primary partition.

4. The method of claim 3 further including the step of providing the secondary partition with a record session table, storing start and end times of recording session, and index start and end entries of the index table to indicate location in the index table of selected audio.

5,396,371

5

5. In a system for processing audio having an interface for receiving audio from an audio source, a digital signal processor in communication with the interface for compressing the audio signals, a controller in communication with the digital signal processor for receiving audio therefrom and arranging data in a prescribed order, a supervisor in communication with said controller accessing data from said system, and a buffer in communication with the controller for receiving arranged audio from the controller, the improvement comprising:

- a digital audio tape drive unit in communication with the buffer for receiving arranged audio data from the buffer,
- a random access storage device, and
- a pair of pointers providing communication between said buffer and random storage device, the first of said pointers operative for transmitting audio data to said random access storage device from said buffer and the second of said pointers

6

being operative to send audio data from said random access storage device to said controller.

6. The system of claim 5 further including a speaker in communication with said controller for playing audio retrieved from said random access storage device.

7. The system of claim 6 wherein said random access storage device has a primary partition for storing recorded audio data and a secondary partition for storing means for locating selected audio data stored on said primary partition, said second pointer being alternately in communication with said first partition and said second partition.

8. An audio data storage device, comprising:

- a random access storage device having a primary partition for storing audio data and a secondary partition for storing means for locating data on said primary partition and a pair of pointers in communication with said random access memory, a first of said pointers being operated to transmit data to said random access storage device and the second of said pointers being operative to retrieve audio data from said random access storage device.

* * * * *

EXHIBIT C



US005819005A

United States Patent [19]**Daly et al.**[11] **Patent Number:** **5,819,005**[45] **Date of Patent:** **Oct. 6, 1998**[54] **MODULAR DIGITAL RECORDING LOGGER**

[75] Inventors: **Daniel F. Daly**, Monroe; **John Henits**, Bethel; **Salvatore J. Morlando**, Easton; **Robert B. Swick**, Stratford; **Keith K. W. Leung**, Monroe; **Constantine P. Messologitis**, Milford, all of Conn.

[73] Assignee: **Dictaphone Corporation**, Stratford, Conn.

[21] Appl. No.: **623,671**

[22] Filed: **Mar. 29, 1996**

Related U.S. Application Data

[63] Continuation of Ser. No. 100,944, Aug. 3, 1993, abandoned.

[51] Int. Cl.⁶ **G10L 3/02; G11B 5/09**

[52] U.S. Cl. **395/2.09; 395/2.79; 395/2.91; 360/48; 379/88**

[58] Field of Search **370/58.2, 62, 84, 370/235, 280; 381/36; 395/2, 2.79, 2.81, 2.87, 2.09, 2.91, 2.94; 360/48, 5, 32; 379/68, 70, 73, 75, 85, 88**

[56] **References Cited****U.S. PATENT DOCUMENTS**

4,260,854 4/1981 Kolodny et al. 360/71
 4,375,083 2/1983 Maxemchuk 364/900
 4,621,357 11/1986 Naiman et al. 370/58.2
 4,630,261 12/1986 Irvin 370/235
 4,679,191 7/1987 Nelson et al. 370/84
 4,829,514 5/1989 Frimmel et al. 370/58

5,001,703 3/1991 Johnson et al. 370/280
 5,121,212 6/1992 Okumura et al. 348/738
 5,129,036 7/1992 Dean et al. 395/2
 5,142,527 8/1992 Barbier et al. 370/62
 5,353,168 10/1994 Crick 360/5
 5,448,420 9/1995 Henits et al. 360/48
 5,511,000 4/1996 Kaloi et al. 364/514 A

FOREIGN PATENT DOCUMENTS

4005027 2/1992 Germany H04M 11/06
 2174330 7/1990 Japan H04L 12/18
 1712964 2/1992 U.S.S.R. G11C 11/40

OTHER PUBLICATIONS

Brochure—VR240 Digital Broadcast Logger (no date).

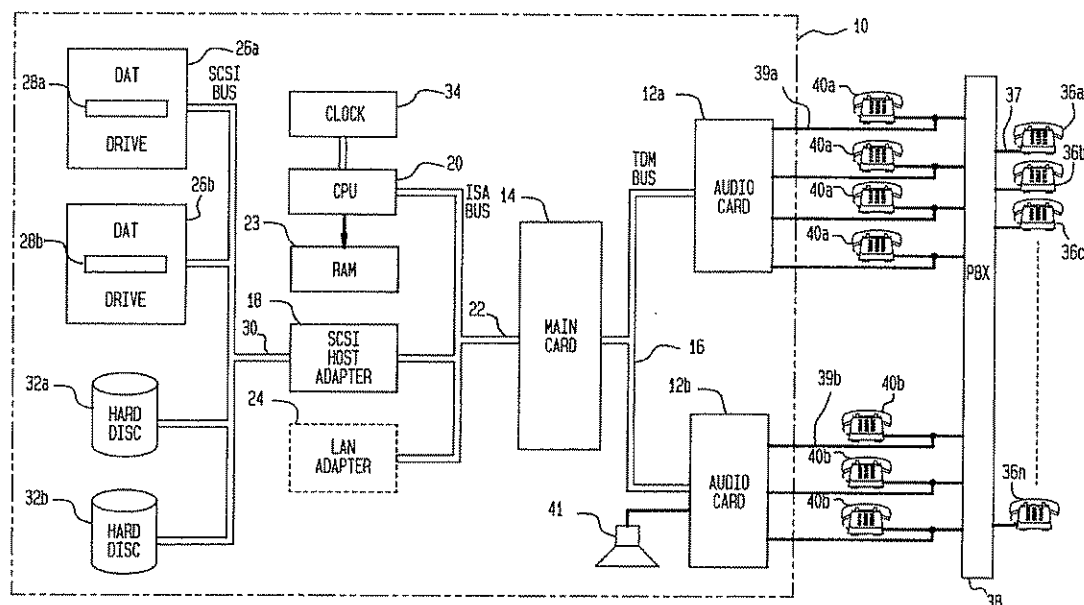
Primary Examiner—Allen R. MacDonald

Assistant Examiner—Richmond Dorvil

Attorney, Agent, or Firm—Curtis, Morris & Safford, P.C.; Gregor N. Neff

[57] **ABSTRACT**

A modular digital recording system that records audio on digital audio tapes provides redundancy and the ability to record audio while listening to portions of audio that had been recorded on the DAT. The system uses a hard disc that receives audio for recording simultaneously with the DAT. When one wishes to listen to a voice message, one can activate the hard disc to listen to a particular message while the DAT is still recording. The system is modular so that the capacity can be expanded as required. The system includes a LAN adapter so that the system can provide networking access.

26 Claims, 3 Drawing Sheets

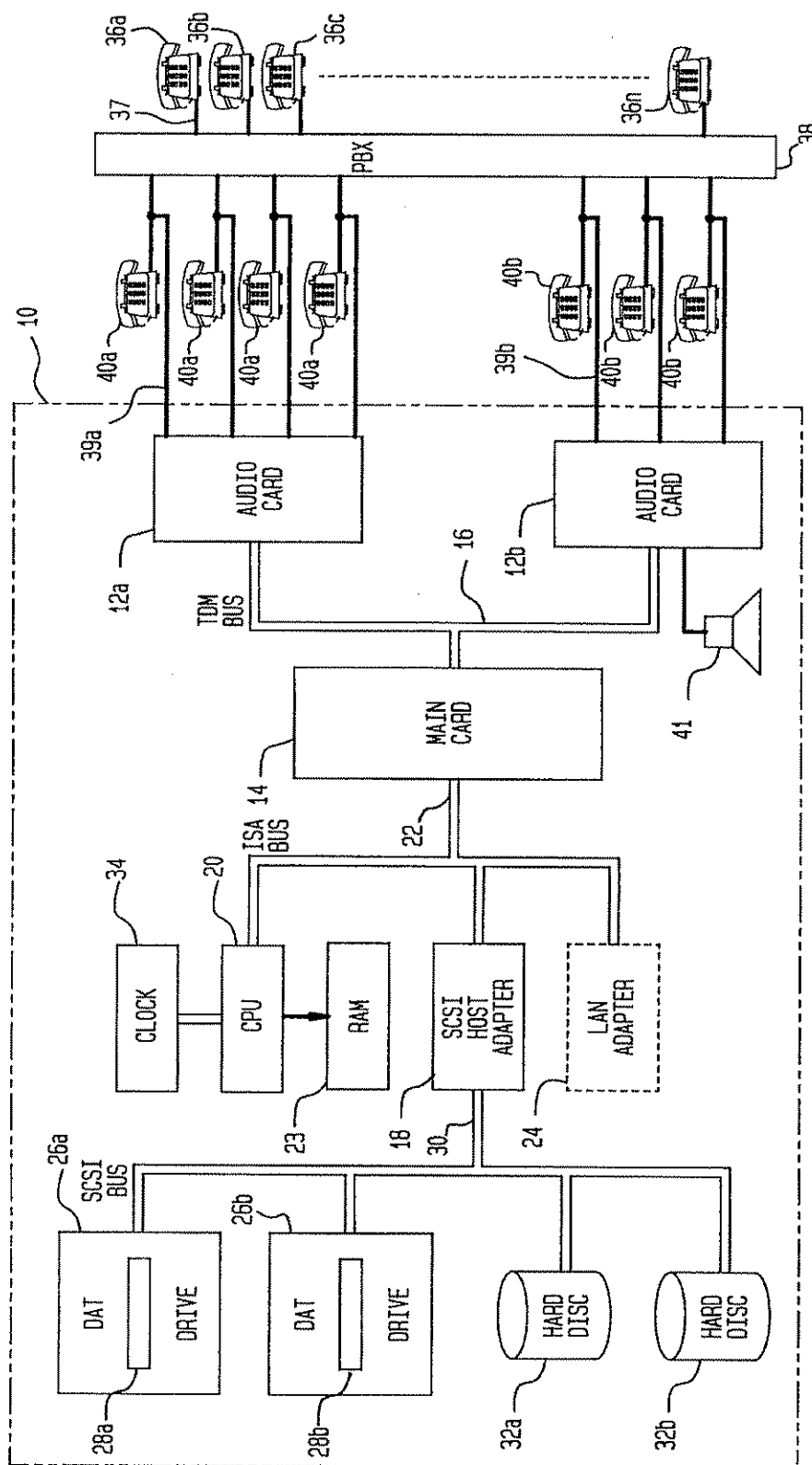
U.S. Patent

Oct. 6, 1998

Sheet 1 of 3

5,819,005

FIG. 1

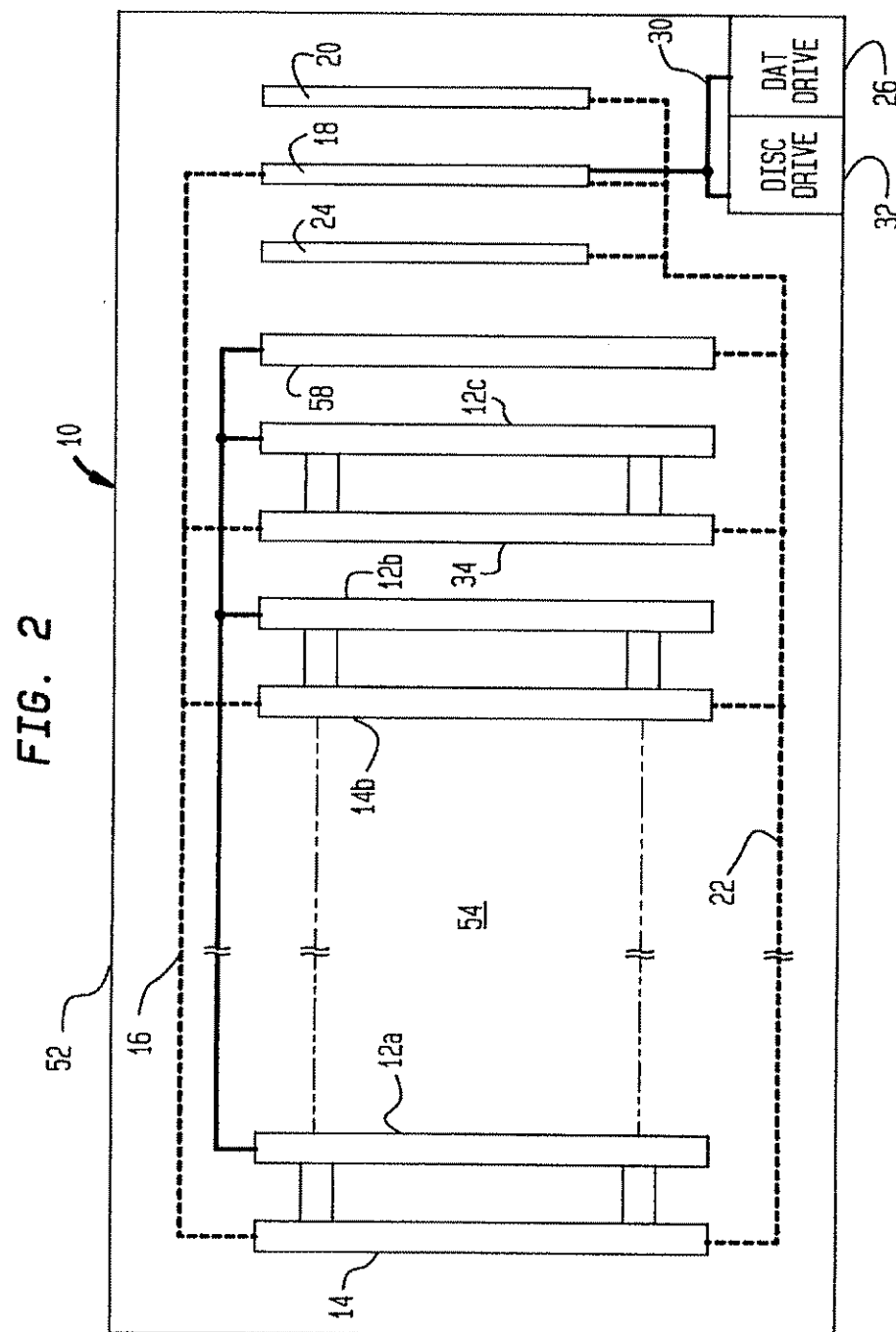


U.S. Patent

Oct. 6, 1998

Sheet 2 of 3

5,819,005



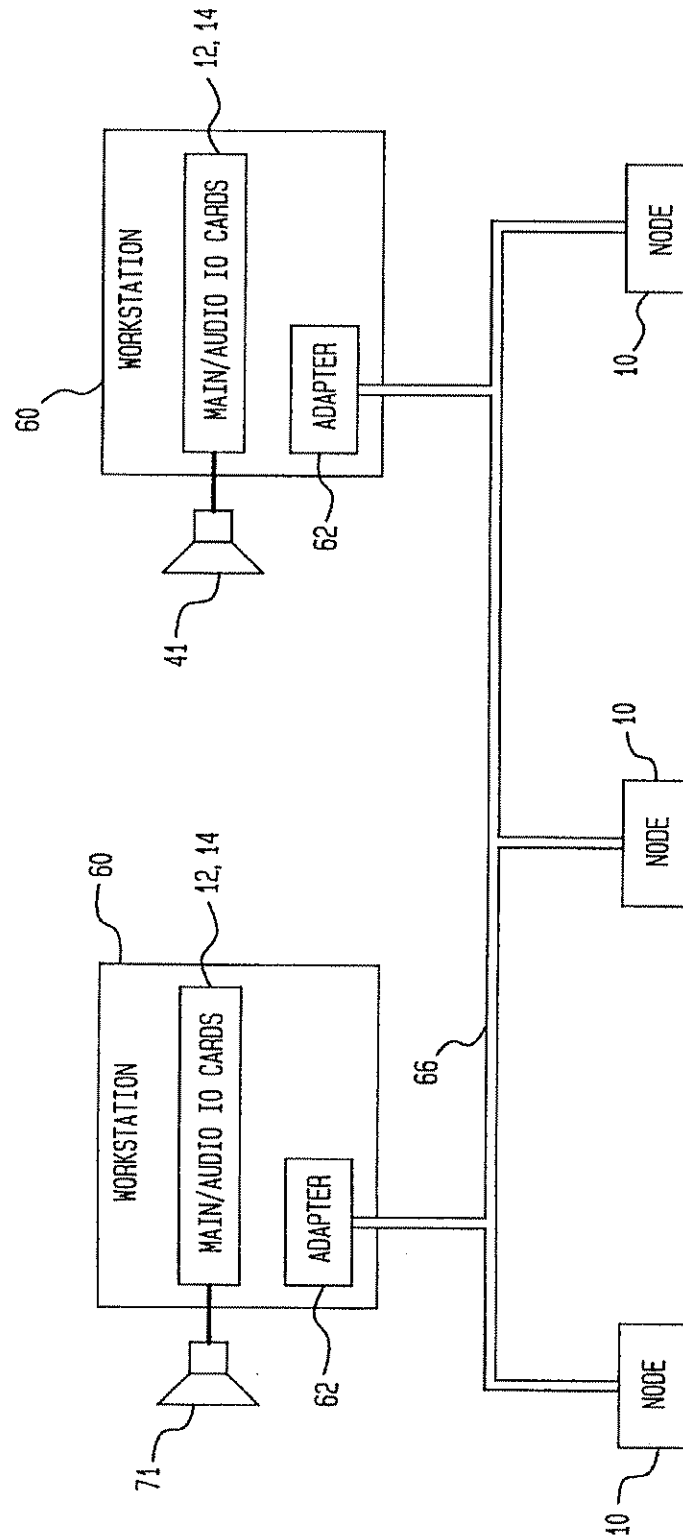
U.S. Patent

Oct. 6, 1998

Sheet 3 of 3

5,819,005

FIG. 3



5,819,005

1

MODULAR DIGITAL RECORDING LOGGER

This is a continuation of U.S. patent application Ser. No. 08/100,944, filed Aug. 3, 1993 now abandoned.

BACKGROUND OF THE INVENTION

In the field of voice processing, there are circumstances in which it is necessary that audio, such as conversations, be recorded and the time when such recordings took place be established. Systems capable of providing this requirement have been commercially available for a long period and are referred to as recording loggers, or loggers for short. Prior systems have worked rather well, but were based for a long time on analog technology. Because of this, the prior logger systems were physically large and the tapes that recorded audio for archival purposes were also large, thus requiring a large amount of storage space.

To overcome these drawbacks of prior analog loggers, digital loggers have recently been developed and offered commercially. Although such digital loggers have advantages over the prior analog loggers, they still have shortcomings in terms of networking expandability and voice capacity. It is an object of the invention to provide a digital logger that overcomes these disadvantages. In addition, it is an object of the invention to provide a digital logger that is modular in construction so that the capacity of such logger can be increased conveniently and economically and software can be upgraded as required.

SUMMARY OF THE INVENTION

The modular digital recording logger of the invention provides advantages not only over prior analog loggers, but over prior digital recording loggers as well. The digital logger of this invention has a basic unit that comprises four primary components, an audio card that monitors audio sources (such as telephones), a main card that processes audio, a host computer that controls the overall operation and memory.

The audio card serves the main functions of communicating with the audio sources, converting received analog signals to digital signals and directing the audio signals through a time division multiplexed (TDM) bus to the main, or application card.

The application card communicates with the audio card through the TDM bus to monitor the status of the audio cards, when there is more than one, and determine which needs service. The application card packages received data, executes speech compression and expansion, performs VOX and performs other functions. The application card is attached to an ISA bus as are a computer, such as a personal computer, a LAN adapter and a SCSI adaptor. The computer stores the operating instructions and supervises and coordinates the activities of the other components of the logger system. The SCSI adaptor is in communication with at least one digital audio tape (DAT) drive and at least one hard disc drive. The system is modular so that the capacity of the system can be expanded as required readily at a minimum cost and software can be modified conveniently as desired. In addition, the LAN adapter allows each logger of the invention to be part of a networked system that include other digital loggers and workstations.

BRIEF DESCRIPTION OF THE DRAWING

FIG. 1 shows a functional block diagram of a modular digital recording logger constructed in accordance with the present invention;

2

FIG. 2 is a top plan schematic view of the components of the logger of FIG. 1 shown in a housing; and

FIG. 3 is a block diagram showing a plurality of the loggers of FIG. 1 connected in a network.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

With reference to FIG. 1, a digital modular recording logger is shown at 10 and includes a pair of audio circuit boards 12a, 12b which will hereinafter be referred to as audio cards. Only two audio cards 12 are shown, but a larger number of audio cards can be included in the system. For example, where thirty two recording channels are required, eight digital audio cards 12 would be required, as it will be described in the description that follows. The audio cards 12a, 12b are in communication with an audio application processing circuit board 14, which hereinafter will be referred to as the main card, through a time division multiplexed (TDM) bus 16. Once more, only one main card 14 is shown and described, but it will be appreciated that where more audio cards are required, (e.g., eight), more main cards also may be needed, (e.g., four), upon the particular architecture of the audio cards and main cards. The main card 14 communicates with a personal computer through an ISA bus 22. The computer has a CPU and RAM 23 and is in communication with a small computer system interface (SCSI) adaptor 18 through the ISA bus 22. The SCSI host adaptor 18 can be any of a number of commercially available adaptors such as an Adaptec 1542 adaptor available from Adaptec Corporation.

The combination of the CPU 20 and RAM 23 can be any personal computer such as an IEEE 996 standard PC/AT.

Also tied to the CPU 20 is a clock 34. The LAN adapter 24 provides the opportunity for network connection, as it will be described hereinafter with reference to FIG. 3.

The SCSI host adaptor 18 communicates with a pair of tape drives 26a, 26b each of which is capable of driving a digital audio tape (DAT) 28a, 28b, respectively. The tape drives 26a, 26b are in communication with the SCSI host adaptor 18 through a SCSI bus 30. Also in communication with the SCSI host adaptor 18 through the SCSI bus 30 are two disc drives 32a, 32b. The number of DAT's 28 and disc drives 26 can vary to customize the system 10 to the operative channel requirements, but where the tapes are redundant, only one disc drive need be used.

A plurality of audio sources such as telephones 36a, 36b, 36c . . . 36n are shown. These represent any telephones outside of the system 10 that are able to communicate with the system 10 through communication lines 37. The system 10 can monitor a number of different types of audio devices, including a private branch exchange (PBX) 38 to which a plurality of telephones 40a and 40b are connected. Other audio sources include radio, central office lines, microphones, speakers and the like.

As shown in FIG. 1, the audio card 12a has four ports and is able to communicate with the lines 39 (channels) connecting the telephones 40a with the PBX 38 so as to receive audio signals therefrom. More specifically, the audio cards 12a, 12b receive audio from two sources, an internal telephone 40 and an external telephone 36. The audio card 12b also has four ports, with three of the ports communicating with communication lines 39b leading to telephones 40b and the PBX 38. One of the ports is connected to a speaker 41, thereby allowing messages to be heard, as it will be described hereinafter. The communication lines 39 transmit audio from the PBX 38 and telephones 40 to the audio cards 12.

5,819,005

3

The audio cards 12 can be of the type shown and described in co-pending patent application Ser. No. 07/815,205 and entitled Audio Circuit Board for a Modular Digital Voice Processing system, filed Dec. 31, 1991, now Ser. No. 102,678, filed Aug. 5, 1993, the disclosure of which is hereby incorporated by reference. The system 10 can be activated in one of two ways, either through the audio card detecting a telephone 40 going off hook, or through a VOX operation controlled by the main card 14 which is activated upon the main card receiving an audio signal from one of the audio cards.

The audio card 12 converts the analog signals received from the communication lines 39 from analog to digital and will transmit the signals onto the TDM bus 16 under control of the main card 14. The main card 14 monitors the status of the audio cards 12a, 12b to see which audio card needs service and upon responding thereto, the audio is sent over the TDM bus 16. The main card 14 can be of the type shown and described in copending application Ser. No. 07/816,404, filed Dec. 31, 1991, now U.S. Pat. No. 5,404,455, and entitled Time Division Multiplexer Chip, the disclosure of which is hereby incorporated by reference. As stated previously, only two audio cards 12 are shown in FIG. 1, but more audio cards can be added to the system as required.

The main card 14 receives the digital audio signals from the audio cards 12 and compresses the data, as for example from 64K bits per second to 13K bits per second of audio and packages the audio into 2048 byte messages. This data is then sent across the ISA bus 22 to the CPU 20 that serves as a traffic director for the entire system 10. The data is stored in the RAM 23 prior to being transmitted to SCSI host adapter 18 and onto the tapes 28 and discs 32 where the data is stored permanently.

Because DAT recording is relatively fast compared to channel data rates i.e., the DATs are capable of receiving data faster than data is digitized by the system, the data is first written into and stored on the RAM 23 and will be transmitted to the DATs 28 at a rate that is compatible with the capability of the DATs. At the same time, data is written into the disc files 32.

The two tapes 26a, 26b, can be used either independently to achieve greater capacity, or simultaneously to achieve redundancy. When used independently, more data can be written to the tapes. When they are used simultaneously, one tape 28a will serve as a backup for the other tape 28b. In this way, if either of the tapes is destroyed for any reason, there is always a backup.

The disc files 32a and 32b duplicate what is recorded on each of the DATs 28. Clearly, if the DATs 28a, 28b are being used in a redundant manner, only one disc 32 is required to record the audio. In fact, one disc 32 can be partitioned with a portion duplicating the data on one DAT 28a and the other portion duplicating the data on a second DAT 28b when the DATs are run sequentially. The primary function served by the disc file 32 is to allow one to listen to previously written recorded data without interfering with the functions of the DATs 28. In addition, the discs 32a, 32b can serve the function of back-ups for the DATs 28a, 28b.

When one wishes to listen to audio that had been recorded at a particular time, this can be accomplished by inputting a request to the CPU 20 requesting that a message on a particular channel and at the given time be delivered over the speaker 41.

The time data is stored on the DATs 28 and discs 32 under control of the CPU 20. An input is made into the system 10, as by a network or serial port connection for the time and

4

channel for a particular message. The host adaptor 18 will search the disc 32 for the address of the specific time and channel and will retrieve the audio from the disc 32. While this is occurring, the DATs 28a, 28b are capable of continuing to record audio.

The audio recovered from the disc 32a will be transmitted to the main card 14 where decompression will take place and the data will be expanded, as for example, from 13K bits per second to 64K bits per second. The data will then be transmitted to the audio card 12b where it will be converted from digital to analog and eventually will be heard over the speaker 41.

FIG. 2 is a top plan schematic view that demonstrates the modularity of the system 10 and how its capacity can be readily expanded as required. A housing 52 has a base 54 that receives the various cards of the system as will be described. One or more main cards 14a, 14b will be received within and supported by the base 54, as will a plurality of audio cards 12a, 12b. Once more, although three audio cards 12 are shown, it will be appreciated that a greater or lesser number can be used and one main card is capable of monitoring and servicing a plurality of audio cards, two as shown in FIG. 1.

Each main card 14 will be connected logically to an audio card 12a, but it is not necessary that the audio card be connected physically to the main card to which it is connected logically because of the TDM function of the system. In addition, an audio card 12c can be physically attached to a dummy card 34 that can provide an electrical connection but no logic. Thus, one main card 14a can be connected logically through the TDM bus 16 to two audio cards 12a, 12c.

The housing base 54 also supports the CPU 20 which is connected with the main cards 14 through the bus 16. The SCSI adaptor 18 is in communication with the main cards and with the CPU 20 through the ISA bus 22. In this way, cards 12a, 14 can be added or eliminated from the system 10 in accordance with requirements thereof because of the time division multiplexing capability of the system. As a main card 14 and audio card 12 are added they will be connected to the appropriate busses and supported by the base 54. The base 54 also supports a clock buffer 58 that provides timing for the TDM bus and provides additional drive for the ISA bus and the components attached thereto.

FIG. 3 is a schematic view showing how the invention can be networked to include a number of systems 10 connected as nodes, and workstations 60. Where a large number of telephone conversations are to take place and recorded, as for an example at a brokerage firm, a large number (e.g. 160) of conversations may need to be recorded. In this instance, one system 10 would not be capable of servicing that many calls.

In FIG. 3, several workstations 60 are connected to a LAN bus 64 and are several of nodes 10. Each workstation is a personal computer 60 with a main card 14, an audio card 12 and a LAN adapter 62. A speaker 41 is attached to each workstation 60. In addition, the number of workstations 60 can be increased so that access to the systems 10 at a number of different locations can be achieved and one would be able to monitor and have access to any audio data as required.

When audio is to be retrieved, the processing will take place in a node 10 as described previously, with the exception that compressed audio is sent by the LAN adapter 24 (FIG. 1) over the LAN bus 64 (FIG. 3) and received by the LAN adapter 62 of the requesting workstation 60. The compressed data is then processed by the main/audio cards, 14, 12 and subsequently audio is heard over the speaker 41.

5,819,005

5

Thus, what has been shown and described is a digital recording logger that is modular in construction, is capable of using digital audio tapes in an effective manner and can be networked to provide a plurality of workstations and nodes.

The above embodiments have been given by way of illustration only, and other embodiments of the invention will be apparent to those skilled in the art from consideration of the detailed description. Accordingly, limitations on the instant invention are to be found only in the claims.

What is claimed is:

1. A modular digital recording logger, comprising:

a housing;

at least two circuit modules in said housing for converting analog voice signals to digital voice signals, each of said circuit modules including at least two terminals for receiving said analog voice signals, each of said terminals being capable of receiving said analog voice signals for recording a two-way conversation;

a circuit in said housing for compressing said digital voice signals received from each of said circuit modules to provide compressed voice data;

a first bus in said housing for providing communication between said circuit module and said compressing circuit;

a multiplexer circuit in said housing for providing communication between said compressing circuit and said first bus, wherein said multiplexer circuit multiplexes voice signals exchanged between said compressing circuit and said circuit modules on said first bus; and
a digital audio tape (DAT) drive for storing said compressed voice data.

2. The modular digital recording logger of claim 1, further including a clock in communication with said computer.

3. The modular digital recording logger of claim 1, further including a speaker in communication with at least one circuit module.

4. The modular digital recording logger of claim 1, further comprising a hard disk drive in said housing for storing and reproducing said compressed voice data.

5. The modular digital recording logger of claim 4, further comprising: a computer in said housing for operating said DAT drive and/or said hard disk drive to store and reproduce said digital voice signals; and

a second bus in said housing for connecting said computer to said hard disk drive and said DAT drive.

6. The modular digital recording logger of claim 1, wherein said first bus is a time division multiplexing (TDM) bus and said multiplexer circuit is a time division multiplexer circuit.

7. The modular digital recording logger of claim 1, wherein said second bus is a small computer system interface (SCSI) bus and further comprising a SCSI adapter for connecting said computer to said SCSI bus.

8. The modular digital recording logger of claim 1, wherein said compressing circuit is a processor.

9. The modular digital recording logger of claim 8, further comprising an ISA bus for providing communication between said computer and said processor.

10. The modular digital recording logger of claim 7, further including a random access memory (RAM) for storing said compressed voice data before it is transmitted to the SCSI adapter.

11. A network system of modular digital recording loggers, comprising:

at least two digital recording loggers for logging voice conversations, each of said recording loggers comprising:

6

a housing;

a circuit in said housing for converting analog voice signals to and from digital voice signals, said circuit modules including at least two terminals for receiving said analog voice signals, and wherein each of said terminals is capable of receiving said analog voice signals for recording a two-way conversation, a circuit in said housing for compressing said digital voice signals received from each of said circuit modules to provide compressed voice data,

a first bus in said housing for providing communication between said circuit module and said compressing circuit,

a multiplexer circuit in said housing for providing communication between said compressing circuit and said first bus, wherein said multiplexer circuit multiplexes voice signals exchanged between said compressing circuit and said circuit modules on said first bus,

a digital audio tape (DAT) drive for storing said compressed voice data,

a hard disk drive in said housing for storing and reproducing said compressed voice data,

a first computer in said housing for operating said DAT drive and/or said hard disk drive to store and reproduce said digital voice signals, and

a second bus in said housing for connecting said computer to said hard disk drive and said DAT drive;

a second computer for processing compressed digital voice signals received from each of said recording loggers; and

a third bus connecting each of said recording loggers to said second computer.

12. The network system of claim 11, further comprising a clock in communication with said first computer.

13. The network system of claim 11, wherein said third bus is a local area network (LAN) bus.

14. The network system of claim 13, wherein said second computer and each of said recording loggers further include a LAN adapter for providing connection to said LAN bus.

15. The network system of claim 11, wherein said first bus is a time division multiplexed (TDM) bus and said multiplexer circuit is a time division multiplexer circuit.

16. The network system of claim 11, wherein said second bus is a small computer system interface (SCSI) bus and further comprising a SCSI adapter for connecting said first computer to said SCSI bus.

17. The network system of claim 16, further comprising a random access memory (RAM) for storing said compressed voice data before it is transmitted to the SCSI adapter.

18. The network system of claim 11, wherein said compressing circuit is a processor.

19. The network system of claim 18, further comprising an ISA bus for providing communication between said first computer and said processor.

20. The network system of claim 11, wherein said second computer is a workstation.

21. The network system of claim 11, further comprising a speaker in communication with said second computer for reproducing said analog voice signals.

22. A method of manufacturing a modular digital recording logger, comprising the steps of:

selecting a number of circuit modules for converting analog voice signals to and from digital voice signals, each of said circuit modules including at least two terminals for receiving said analog voice signals, and

5,819,005

7

wherein each of said terminals is capable of receiving said analog voice signals for recording a two-way conversation;

installing said selected number of said circuit modules in a housing; 5

installing a circuit in said housing for compressing said digital voice signals received from each of said circuit modules to provide compressed voice data;

installing a first bus in said housing for providing communication between said circuit module and said compressing circuit; 10

installing a multiplexer circuit in said housing for providing communication between said compressing circuit and said first bus, wherein said multiplexer circuit multiplexes voice signals exchanged between said compressing circuit and said circuit modules on said first bus; and 15

installing a digital audio tape (DAT) drive in said housing for storing and reproducing said compressed voice data. 20

23. The method of claim 22, further comprising the steps of connecting to said comprising circuit a hard disk drive for storing and reproducing said compressed voice data.

24. A method of networking a plurality of digital recording loggers, comprising the step of: 25

selecting a number of modular digital recording loggers for logging voice conversations, each of said recording loggers comprising:

a housing; 30

a circuit in said housing for converting analog voice signals to and from digital voice signals, said circuit including a plurality of terminals for receiving said analog voice signals, and wherein each of said ter-

8

minals is capable of receiving said analog voice signals for recording a two-way conversation,

a circuit in said housing for compressing said digital voice signals received from each of said circuit modules to provide compressed voice data,

a first bus in said housing for providing communication between said circuit module and said compressing circuit,

a multiplexer circuit in said housing for providing communication between said processor and said first bus, wherein said multiplexer circuit multiplexes voice signals exchanged between said compressing circuit and said circuit modules on said first bus,

a digital audio tape (DAT) drive for storing and reproducing said compressed voice data,

a hard disk drive for storing and reproducing said compressed voice data,

a first computer in said housing for operating said DAT drive and/or said hard disk drive to store and reproduce said digital voice signals, and

a second bus in said housing for connecting said computer to said hard disk drive and said DAT drive;

installing said selected number of said recording loggers; installing a second computer for processing compressed digital voice signals received from each of said recording loggers; and

installing a third bus connecting each of said recording loggers to said second computer.

25. The method of claim 24, wherein said third bus is a local area network (LAN) bus.

26. The method of claim 25, wherein said second computer and each of said recording loggers further include a LAN adapter for providing connection to said LAN bus.

* * * * *

EXHIBIT D



US006249570B1

(12) **United States Patent**
Glowny et al.

(10) Patent No.: **US 6,249,570 B1**
(45) Date of Patent: **Jun. 19, 2001**

(54) **SYSTEM AND METHOD FOR RECORDING
AND STORING TELEPHONE CALL
INFORMATION**

(76) Inventors: **David A. Glowny**, 53 Candlewood Rd.,
Milford, CT (US) 06460; **Phil Min Ni**,
3 Fairview Dr., Apt. 2, Danbury, CT
(US) 06810; **John E. Richter**, 118
Shelton Rd., Trumbull, CT (US) 06611

(*) Notice: Subject to any disclaimer, the term of this
patent is extended or adjusted under 35
U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/328,299**

(22) Filed: **Jun. 8, 1999**

(51) Int. Cl.⁷ **H04M 1/64**

(52) U.S. Cl. **379/88.22; 379/111**

(58) Field of Search 379/67.1, 68, 88.09,
379/88.11, 88.22, 88.25, 93.12, 93.17, 93.23,
111, 112, 113, 116, 265, 266, 267, 309;
360/5, 6, 55

(56) **References Cited**

U.S. PATENT DOCUMENTS

3,104,284 9/1963 French et al. .
3,369,077 2/1968 French et al. .
3,723,667 3/1973 Park, Jr. et al. 179/100.1
3,727,203 4/1973 Crossman 340/174.1 J
4,130,739 12/1978 Patten 179/100.1
4,199,820 4/1980 Ohtake et al. 365/234
4,260,854 4/1981 Kolodny et al. 179/6.09
4,298,954 11/1981 Bigelow et al. 364/900
4,435,832 3/1984 Asada et al. 381/34
4,442,485 4/1984 Ota et al. 364/200
4,542,427 9/1985 Nagai 360/72.1
4,549,047 10/1985 Brian et al. 179/18 B
4,602,331 7/1986 Sheth 364/200
4,621,357 11/1986 Naiman et al. 370/58
4,630,261 12/1986 Irvin 370/81
4,631,746 12/1986 Bergeron et al. 381/35
4,679,191 7/1987 Nelson et al. 370/84

4,686,587 8/1987 Hipp et al. 360/74.2
4,688,117 8/1987 Dwyer et al. 360/72.3
4,692,819 9/1987 Steele 360/72.1
4,709,390 11/1987 Atal et al. 381/51
4,785,408 11/1988 Britton et al. 364/513.5
4,785,473 11/1988 Pfeiffer et al. 379/89
4,799,144 1/1989 Parruck et al. 364/200
4,799,217 1/1989 Fang 370/68.1
4,811,131 3/1989 Sander et al. 360/74.4
4,811,376 3/1989 Davis et al. 379/57
4,827,461 5/1989 Sander 369/7
4,829,514 5/1989 Frimmel, Jr. et al. 370/58
4,835,630 5/1989 Freer 360/69
4,841,387 6/1989 Rindfuss 360/721
4,851,937 7/1989 Sander 360/69
4,853,952 8/1989 Jachmann et al. 379/88
4,864,620 9/1989 Bialick 381/34
4,873,589 10/1989 Inazawa wt al. 360/53
4,890,325 12/1989 Taniguchi et al. 381/34
4,891,835 1/1990 Leung et al. 379/88
4,893,197 1/1990 Howells et al. 360/8

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

0 544 626 A2 8/1993 (EP) H04Q/11/04
0 437 515 B1 1/1998 (EP) H04M/3/42
0 822 696 A2 2/1998 (EP) H04N/3/36
2 310 102 8/1997 (GB) H04N/1/21

* cited by examiner

Primary Examiner—Scott L. Weaver

(57) **ABSTRACT**

A system and method for monitoring a telephone switching environment. In a preferred embodiment the system and method identify telephone call segments that relate to one telephone call and construct a data representation of a lifetime of the telephone call, using data regarding telephony events associated with the telephone call segments of the telephone call. The system and method are also capable of using the data representation to display a graphical representation of a lifetime of a telephone call.

8 Claims, 29 Drawing Sheets

AgentId	EXTENSION	LOCATION	START TIME	END TIME	CONNECT REASON	DISCONNECT REASON
A		EXTERNAL	t ₁	t ₇	NORM START	NORM DROP
B	0001	INTERNAL	t ₁	t ₄	NORM START	Xfr AWAY
HOLD		INTERNAL	t ₄	t ₆	Xfr Rec	Xfr AWAY
C	0002	INTERNAL	t ₆	t ₇	Xfr Rec	OTHER PARTY HANGUP

US 6,249,570 B1

Page 2

U.S. PATENT DOCUMENTS

4,907,225	3/1990	Gulick et al.	370/94.1	5,270,877	12/1993	Fukushima et al.	360/48
4,939,595	7/1990	Yoshimoto et al.	360/19.1	5,274,738	12/1993	Daly et al.	395/2
4,975,941	12/1990	Morganstein et al.	379/88	5,283,818	2/1994	Klausner et al.	379/67
4,991,217	2/1991	Garrett et al.	381/43	5,305,375	4/1994	Sagara et al.	379/89
5,001,703	3/1991	Johnson et al.	370/79	5,339,203	8/1994	Henits et al.	360/39
5,031,146	7/1991	Umina et al.	365/189.01	5,353,168	10/1994	Crick	360/5
5,065,428	11/1991	Mitchell et al.	380/23	5,396,371	3/1995	Henits et al.	360/5
5,129,036	7/1992	Dean et al.	395/2	5,404,455	4/1995	Daly et al.	395/275
5,130,975	7/1992	Akata	370/60	5,446,603	8/1995	Henits et al.	360/48
5,142,527	8/1992	Barbier et al.	370/62	5,448,420	9/1995	Henits et al.	360/48
5,163,132	11/1992	DuLac et al.	395/275	5,457,782	10/1995	Daly et al.	395/2
5,179,479	1/1993	Ahn	360/72.1	5,533,103 *	7/1996	Peavey et al.	379/69
5,195,128	3/1993	Knitl	379/67	5,819,005	10/1998	Daly et al.	395/2.09
5,210,851	5/1993	Kato et al.	395/425	5,867,559 *	2/1999	Jorgensen et al.	379/67
5,216,744	6/1993	Alleyn et al.	395/2	5,982,857 *	11/1999	Brady	379/88.19
				6,070,241 *	5/2000	Edwards et al.	379/67 X

U.S. Patent

Jun. 19, 2001

Sheet 1 of 29

US 6,249,570 B1

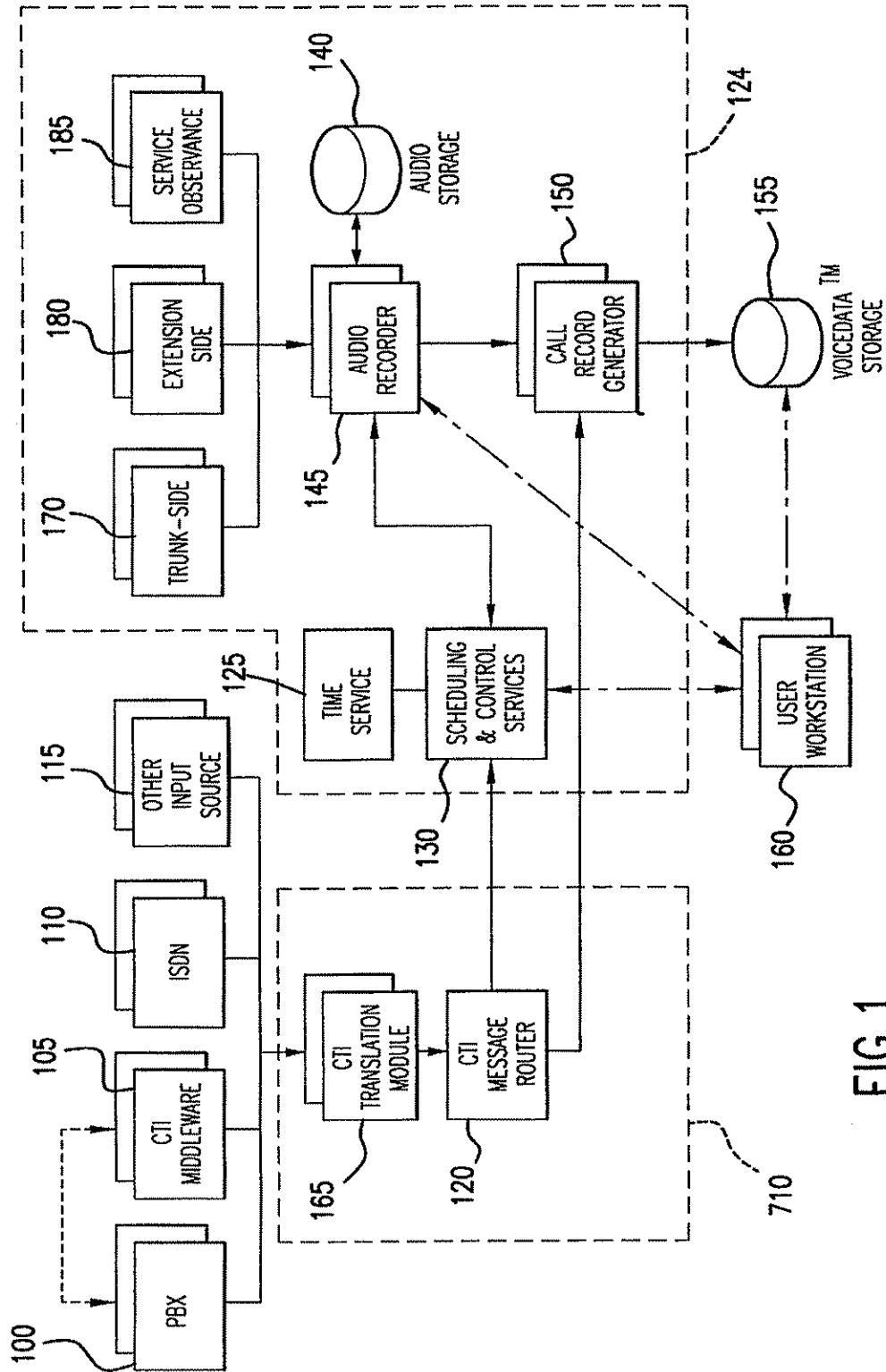


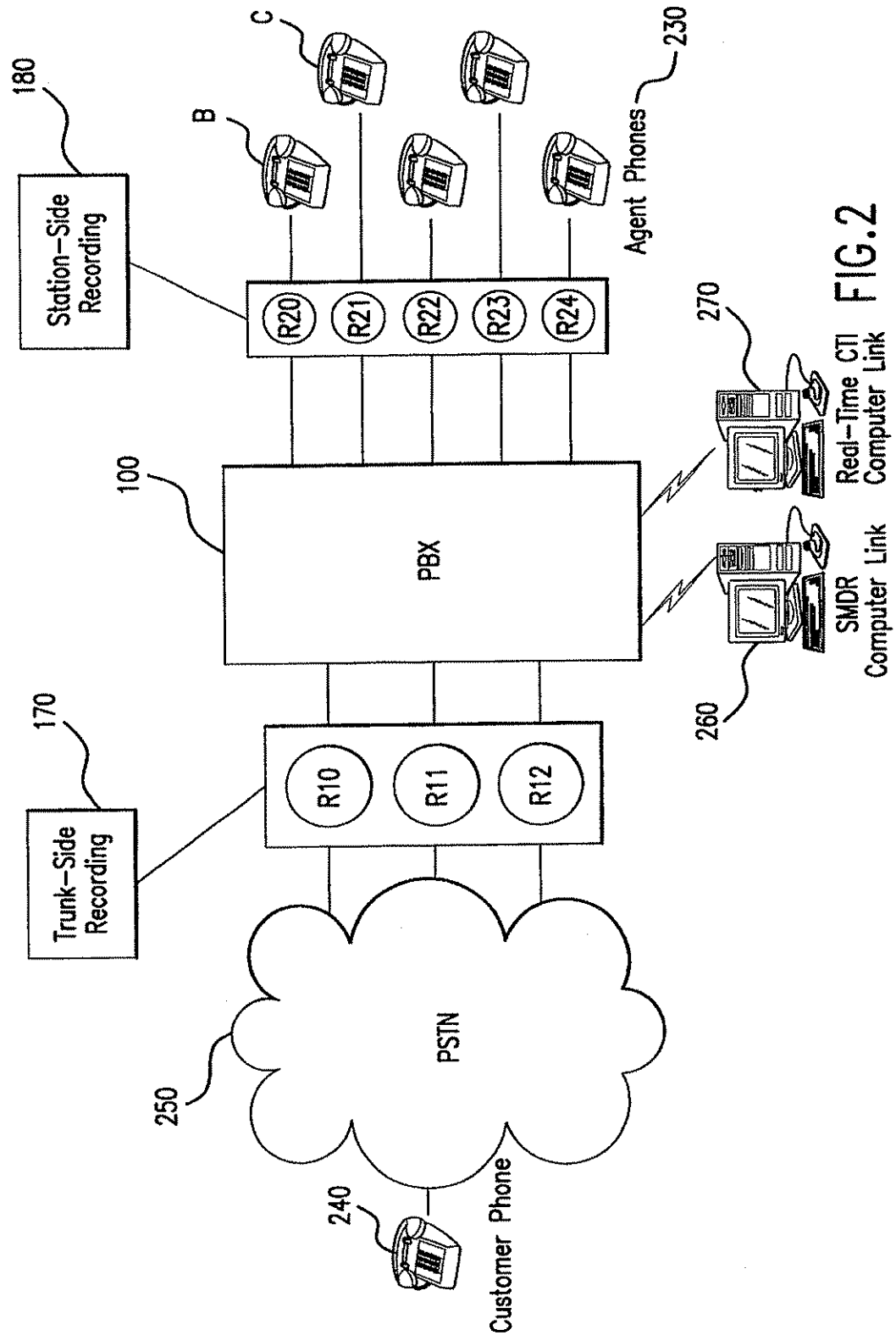
FIG. 1

U.S. Patent

Jun. 19, 2001

Sheet 2 of 29

US 6,249,570 B1



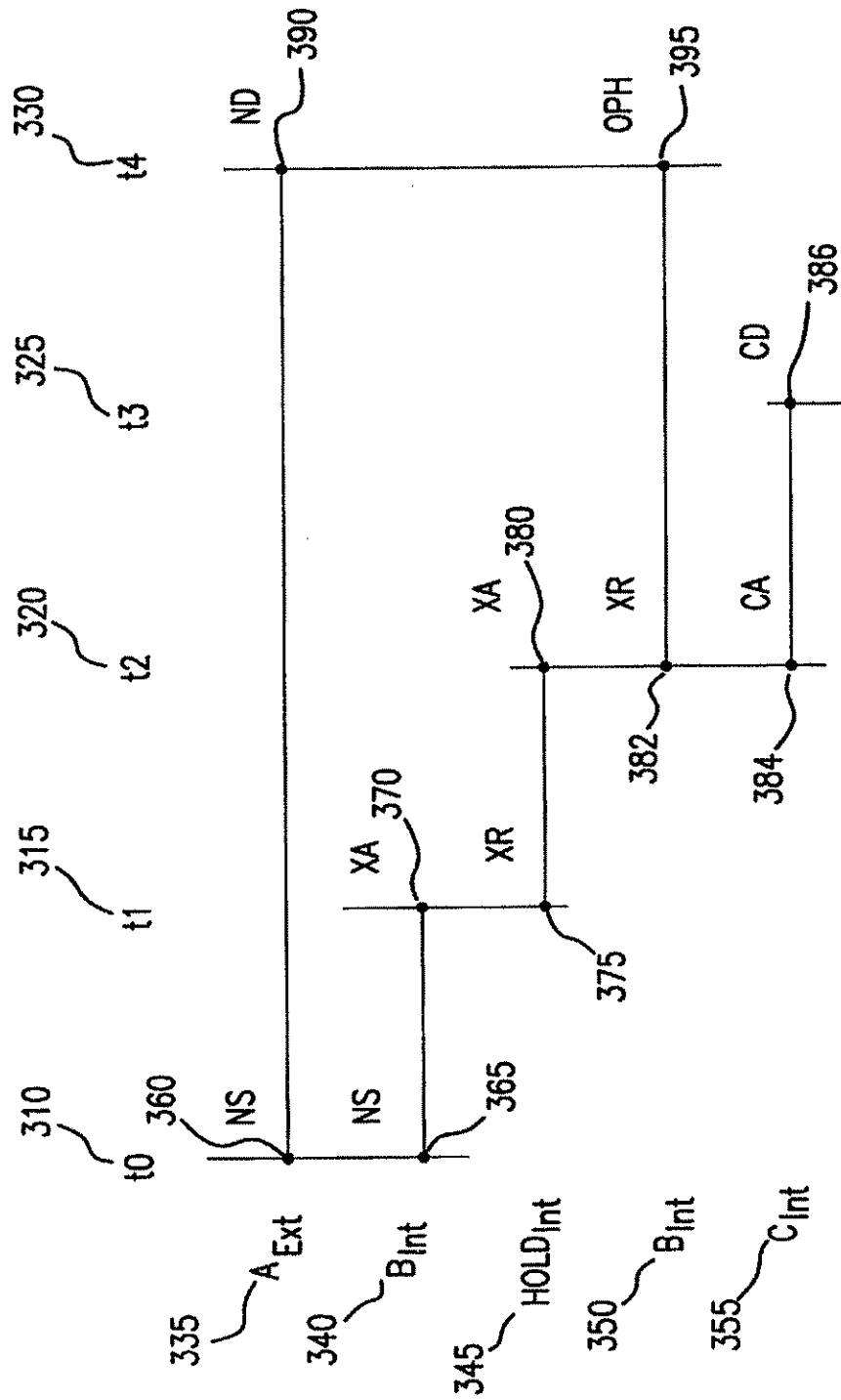


FIG. 3

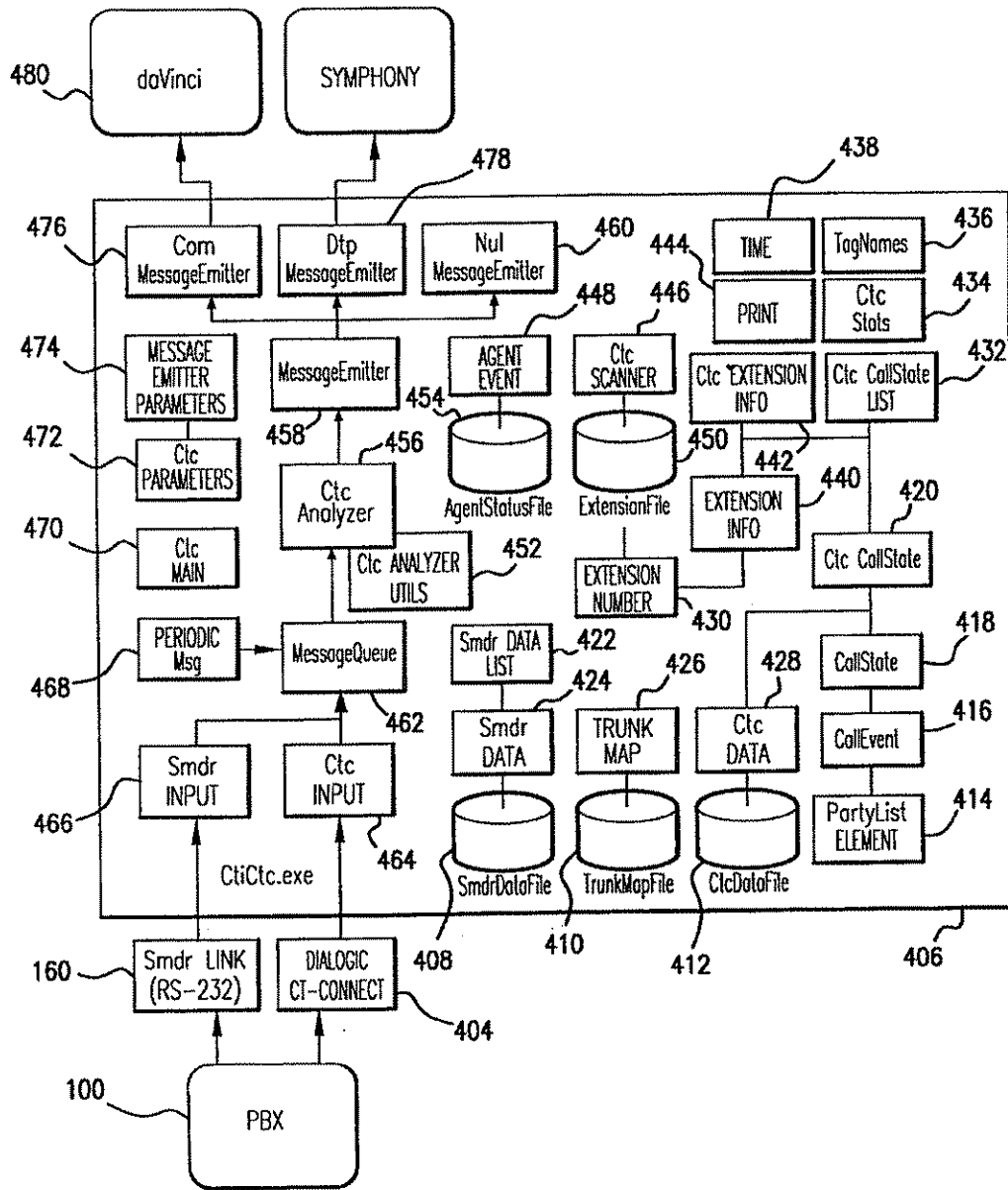


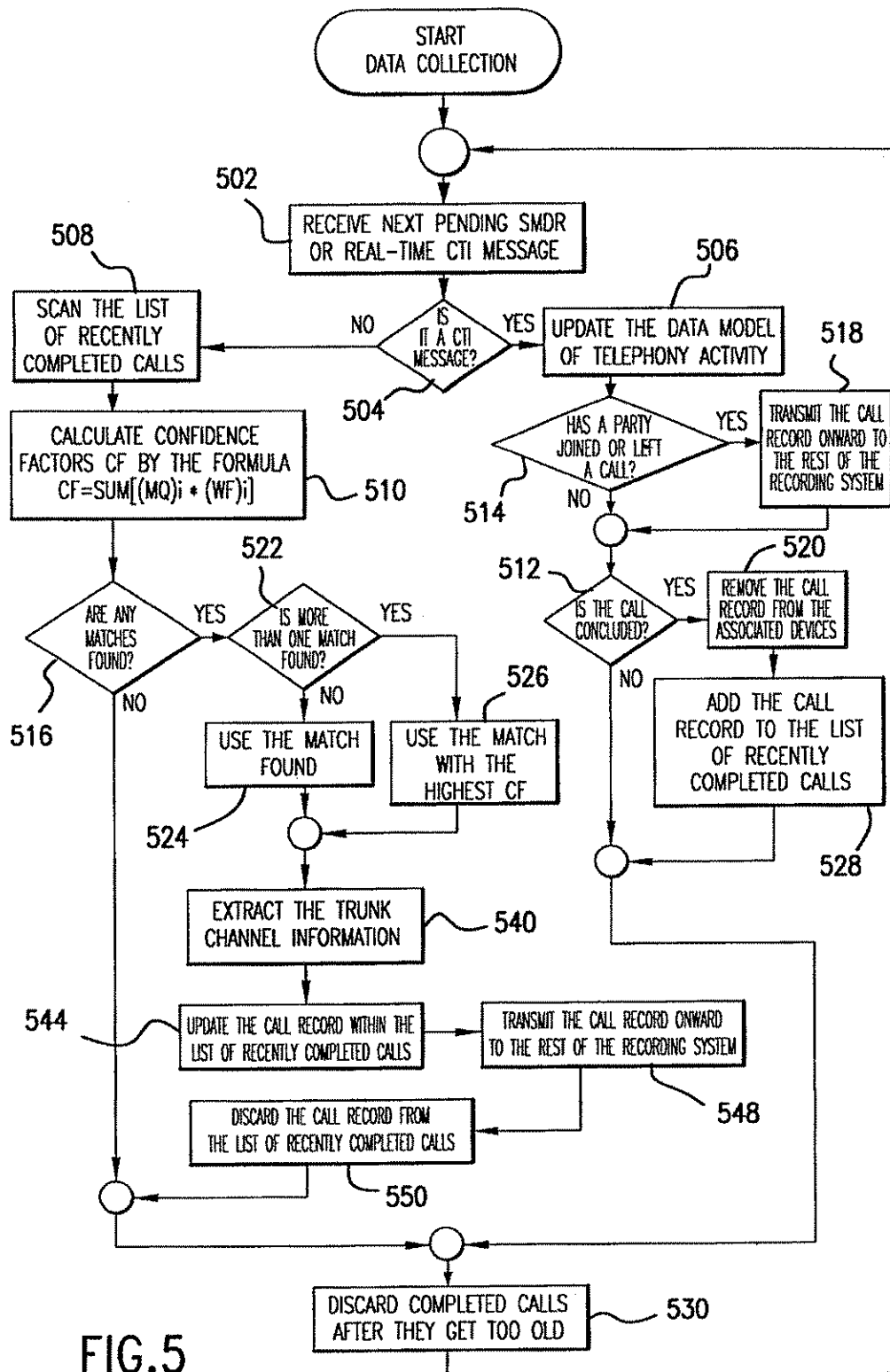
FIG. 4

U.S. Patent

Jun. 19, 2001

Sheet 5 of 29

US 6,249,570 B1



U.S. Patent

Jun. 19, 2001

Sheet 6 of 29

US 6,249,570 B1

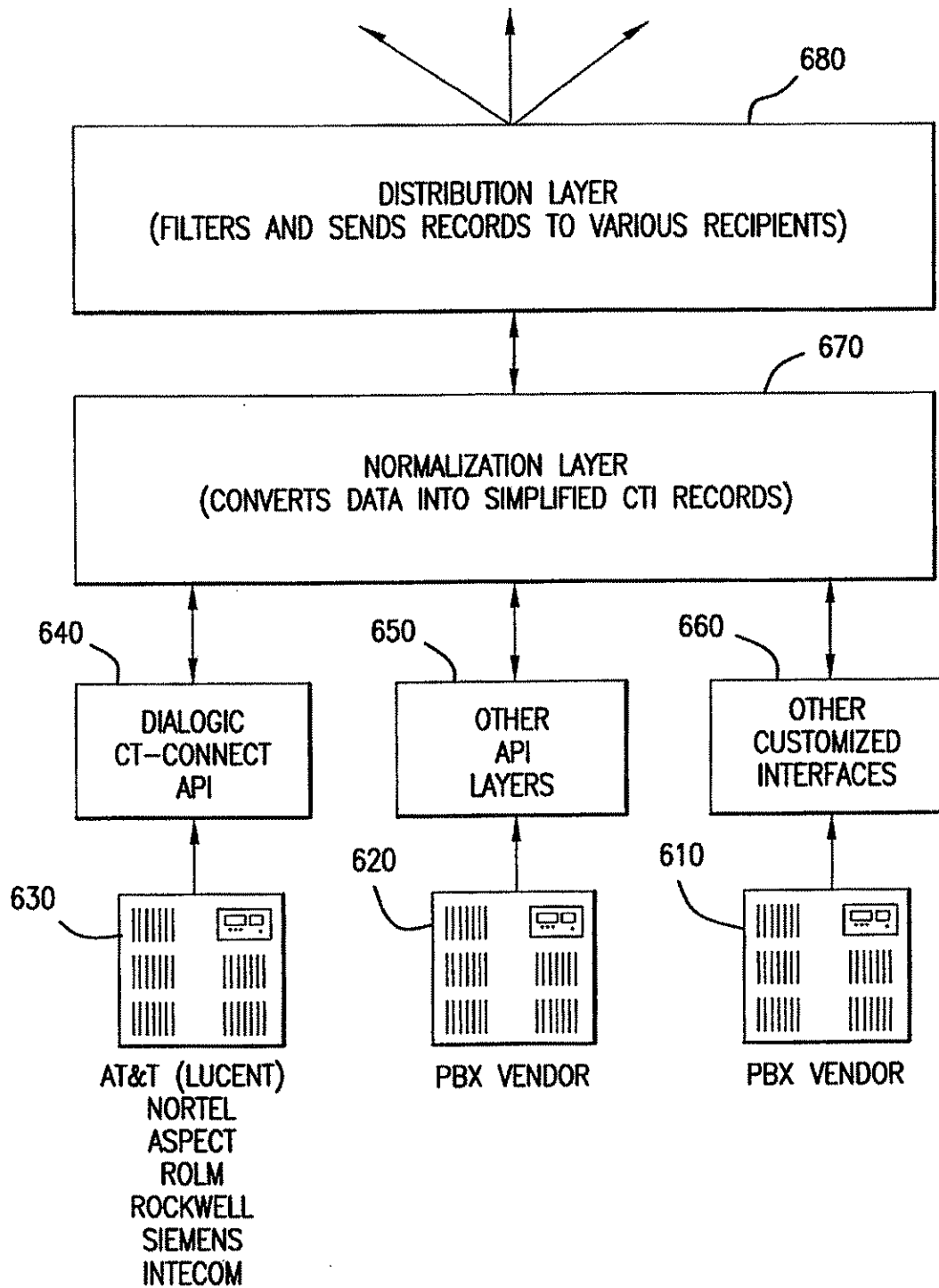


FIG.6

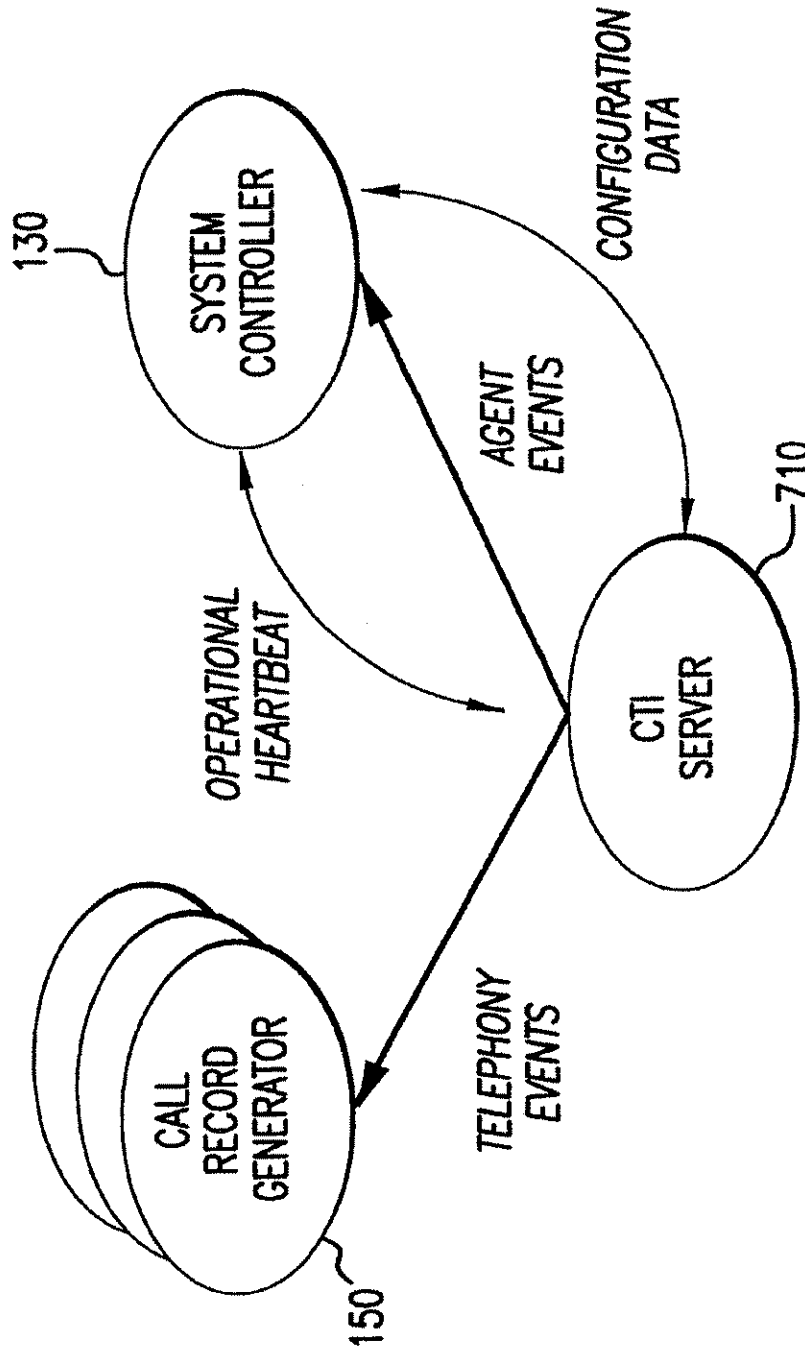


FIG. 7

U.S. Patent

Jun. 19, 2001

Sheet 8 of 29

US 6,249,570 B1

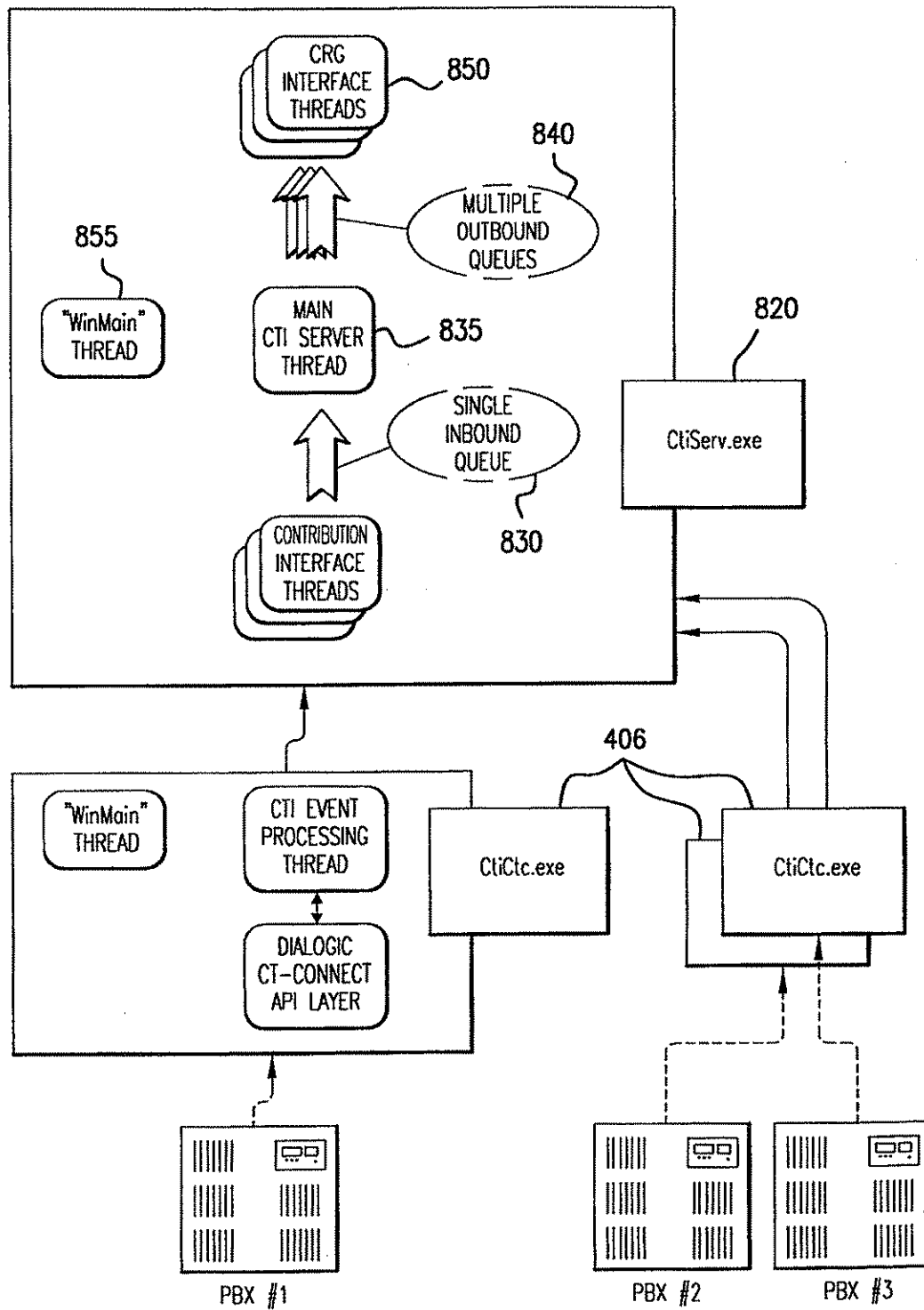


FIG. 8

U.S. Patent

Jun. 19, 2001

Sheet 9 of 29

US 6,249,570 B1

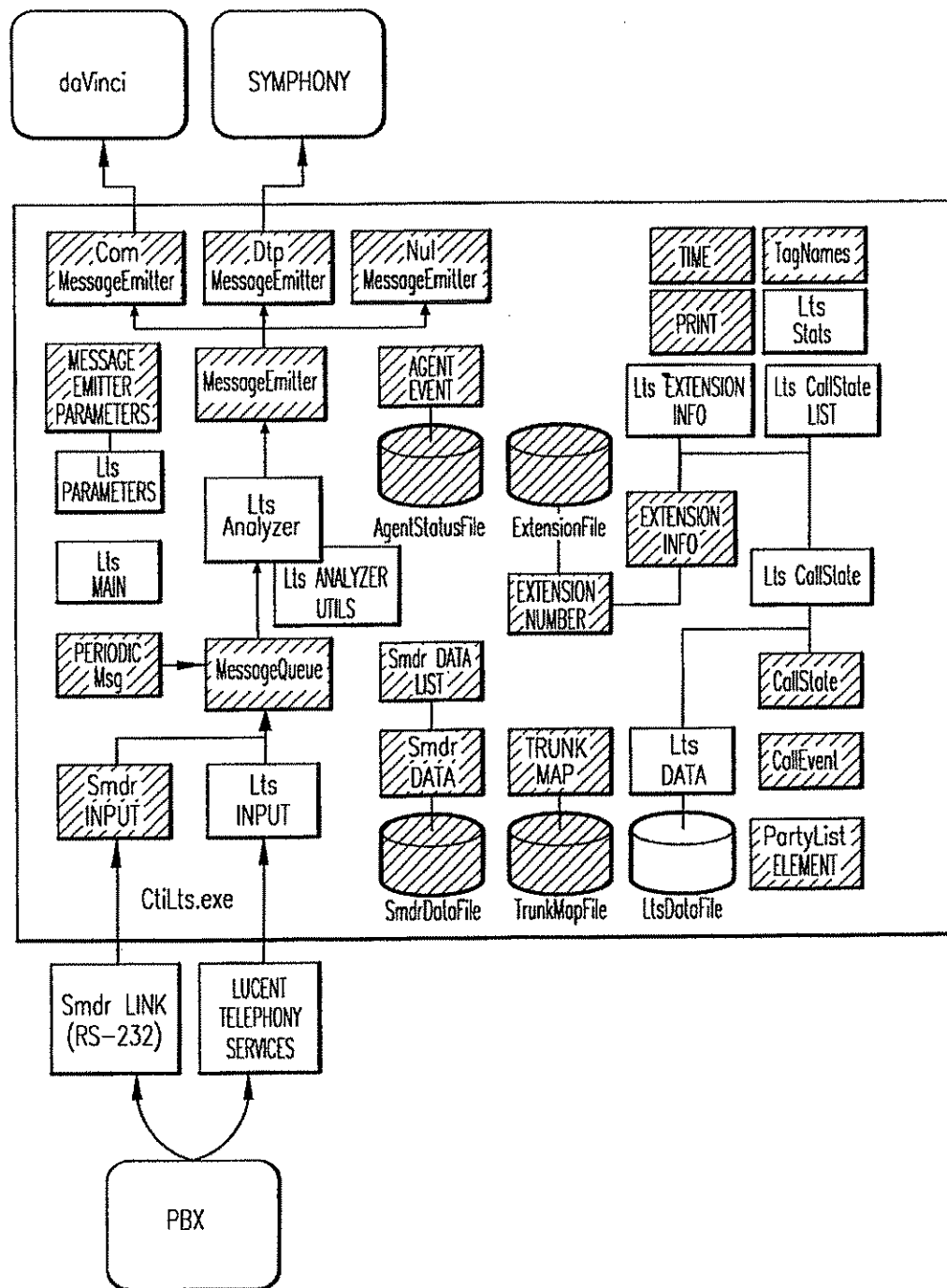
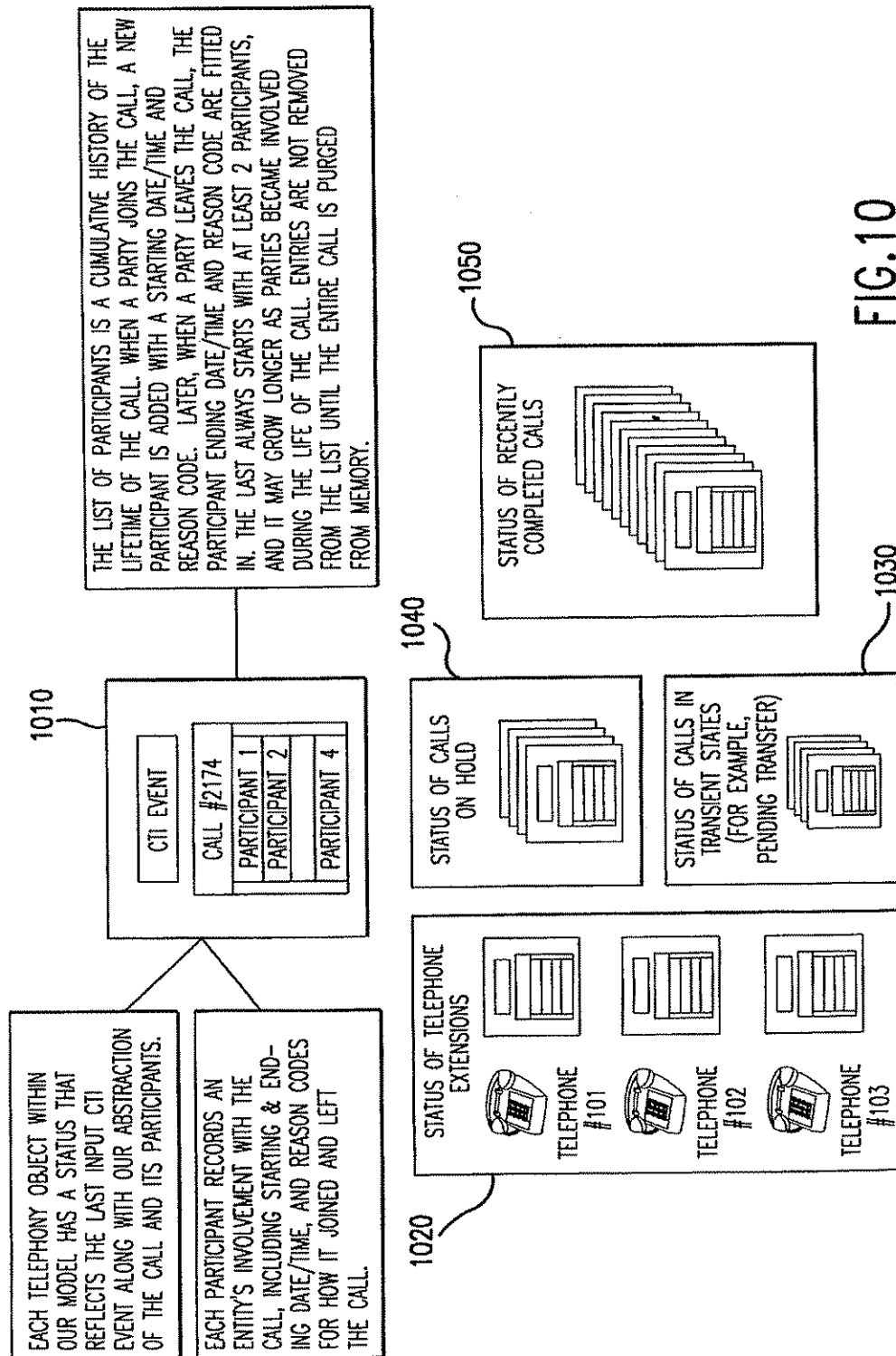


FIG.9



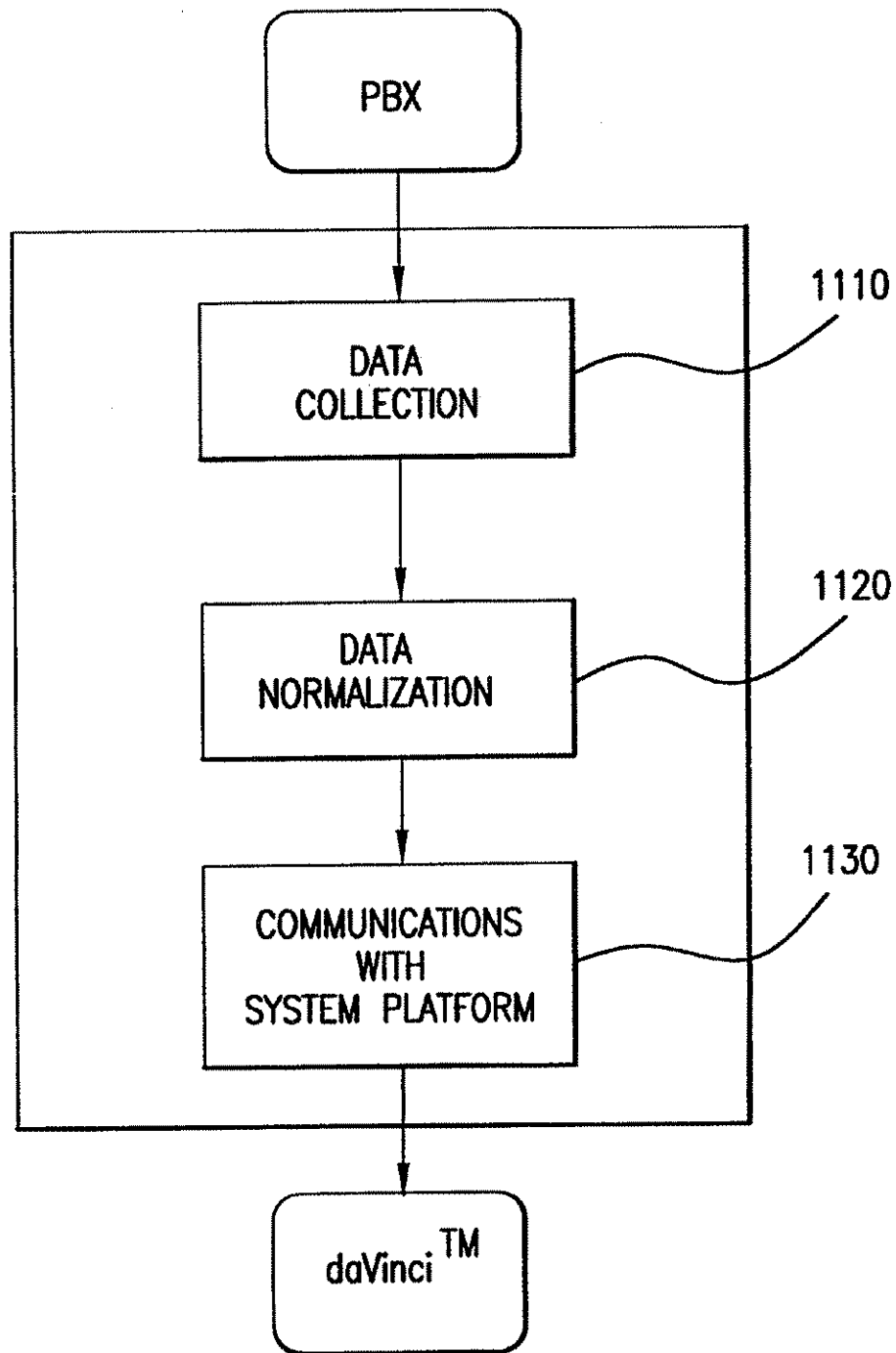


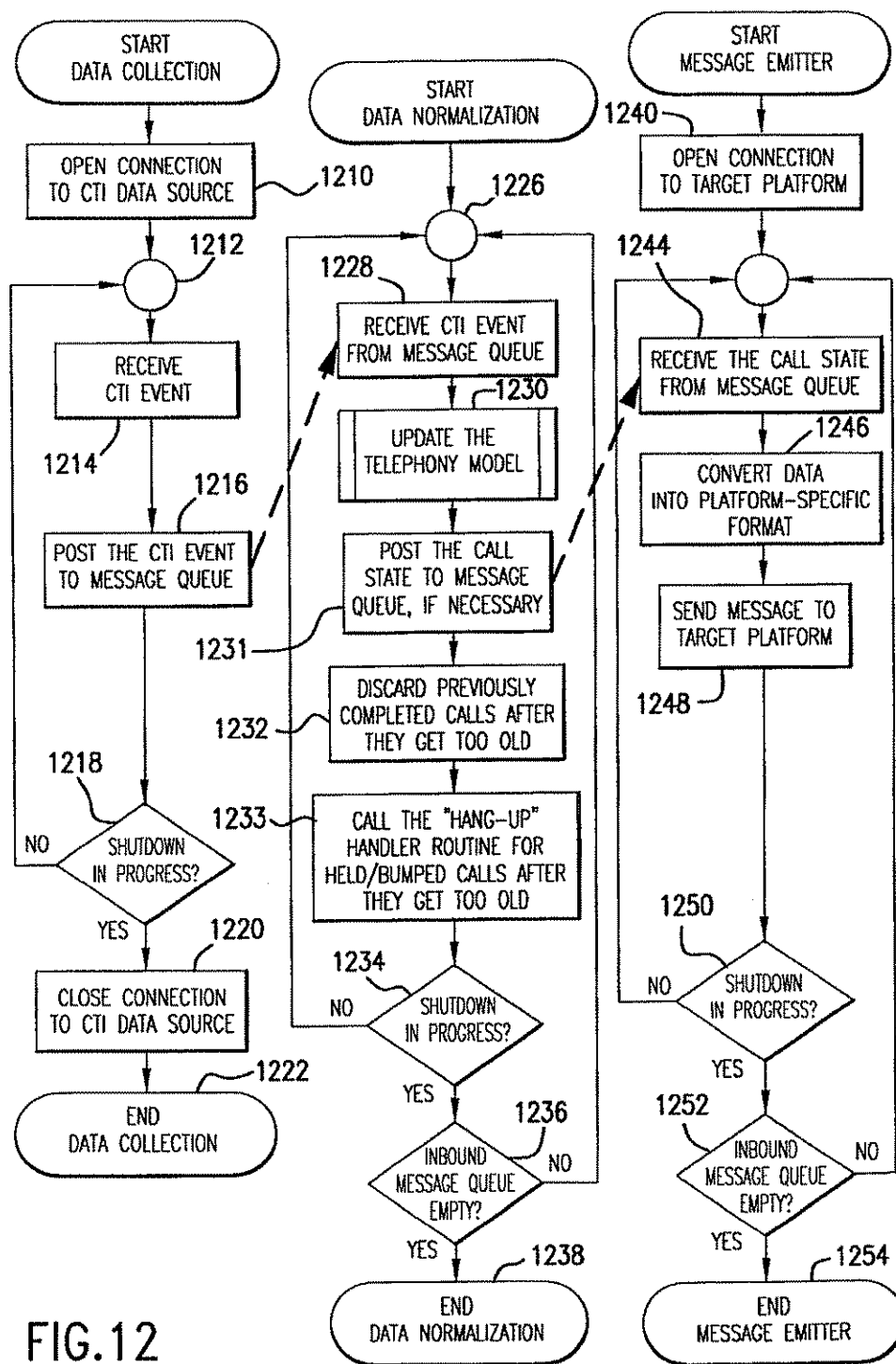
FIG.11

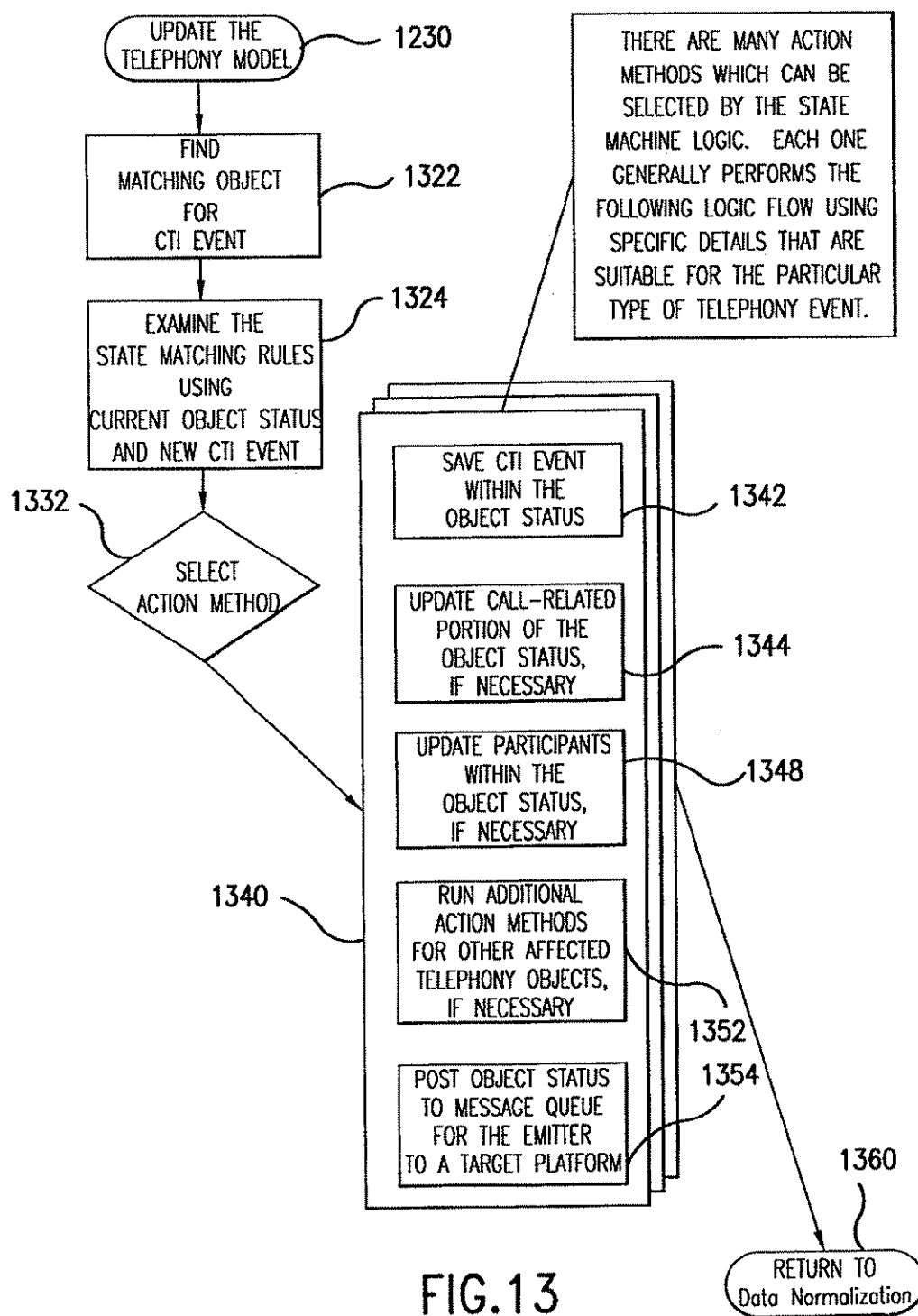
U.S. Patent

Jun. 19, 2001

Sheet 12 of 29

US 6,249,570 B1





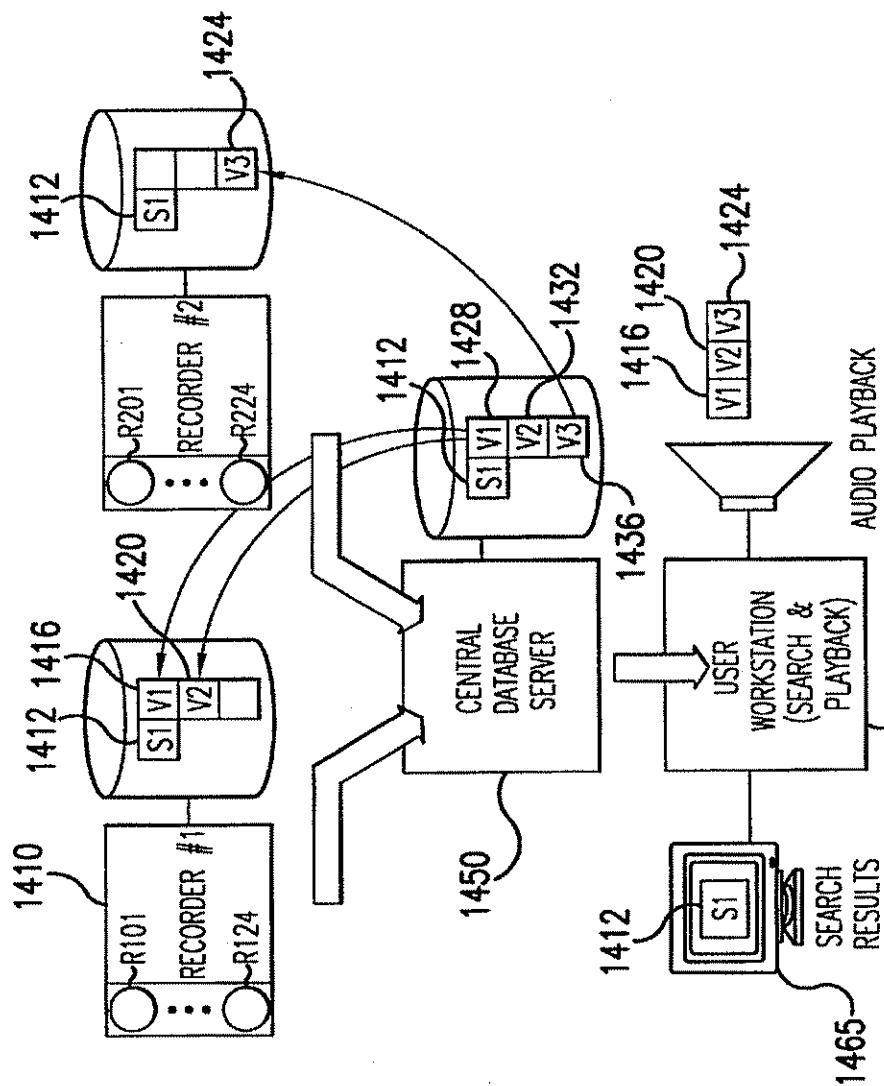


FIG. 14

U.S. Patent

Jun. 19, 2001

Sheet 15 of 29

US 6,249,570 B1

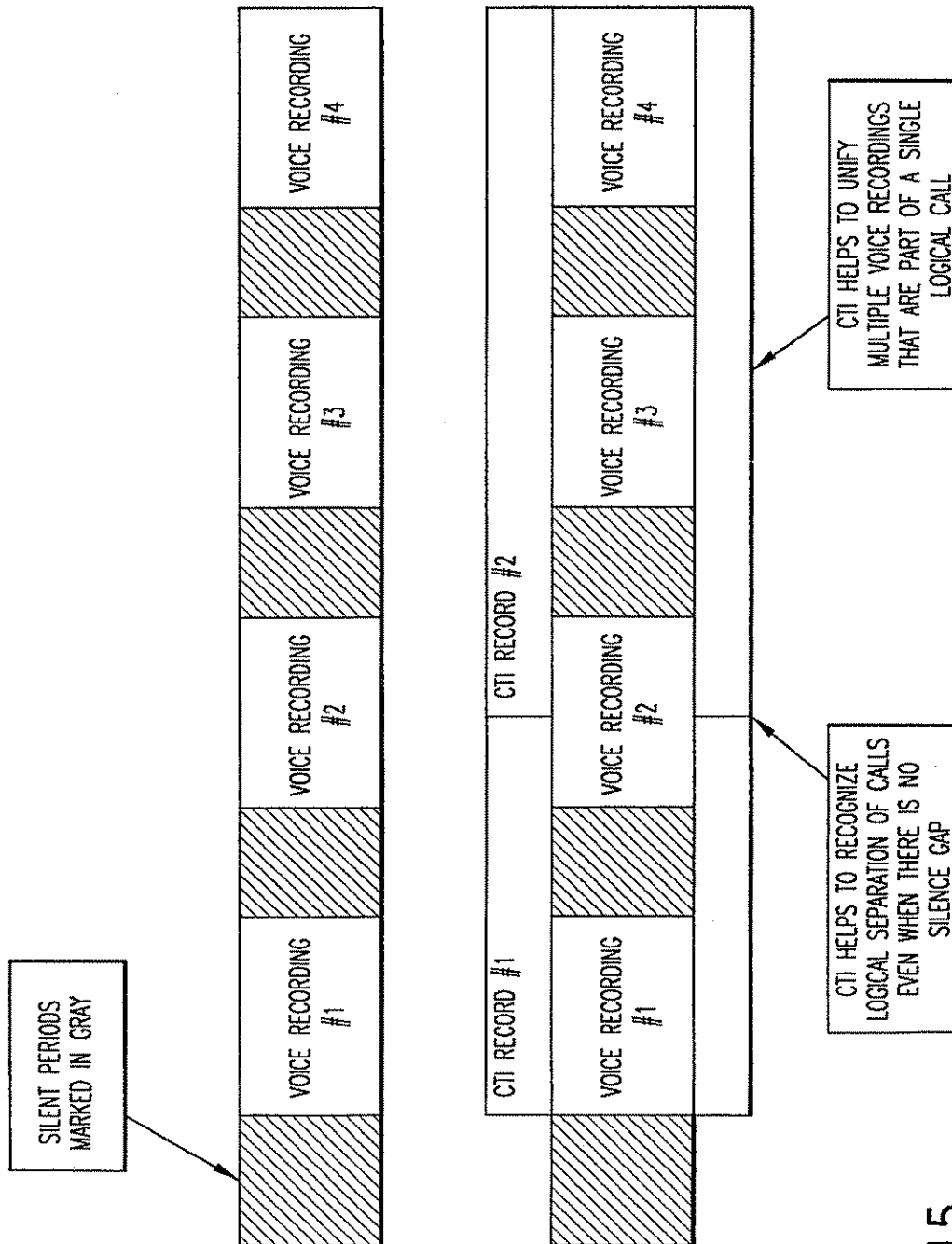


FIG.15

[illegible]

FIG. 16

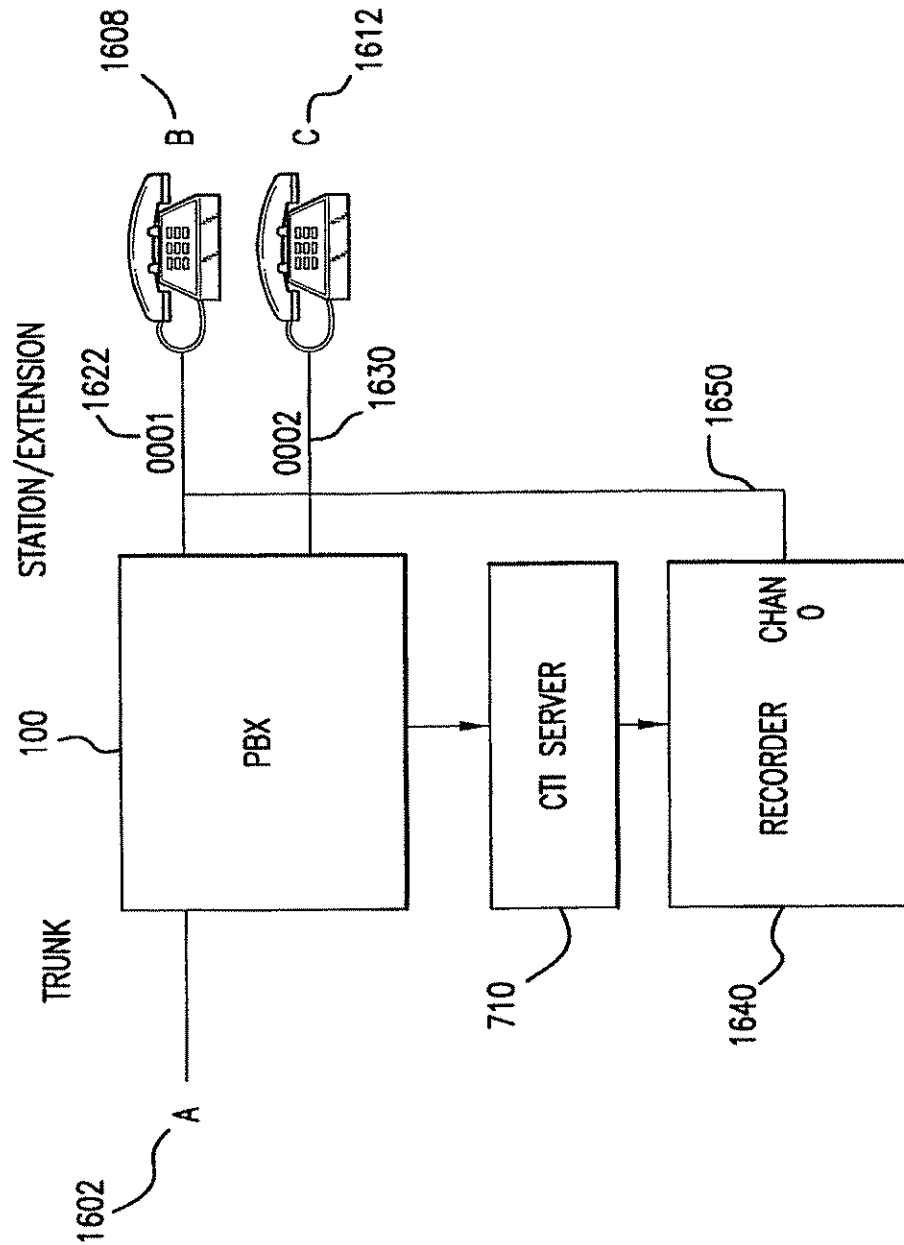


FIG.16A

U.S. Patent

Jun. 19, 2001

Sheet 18 of 29

US 6,249,570 B1

AgentId	EXTENSION	LOCATION	START TIME	END TIME	CONNECT REASON	DISCONNECT REASON
A		EXTERNAL	t_1	t_7	NORM START	NORM DROP
B	0001	INTERNAL	t_1	t_4	NORM START	Xfr AWAY
HOLD		INTERNAL	t_4	t_6	Xfr Rec	Xfr AWAY
C	0002	INTERNAL	t_6	t_7	Xfr Rec	OTHER PARTY HANGUP

FIG.16B

U.S. Patent

Jun. 19, 2001

Sheet 19 of 29

US 6,249,570 B1

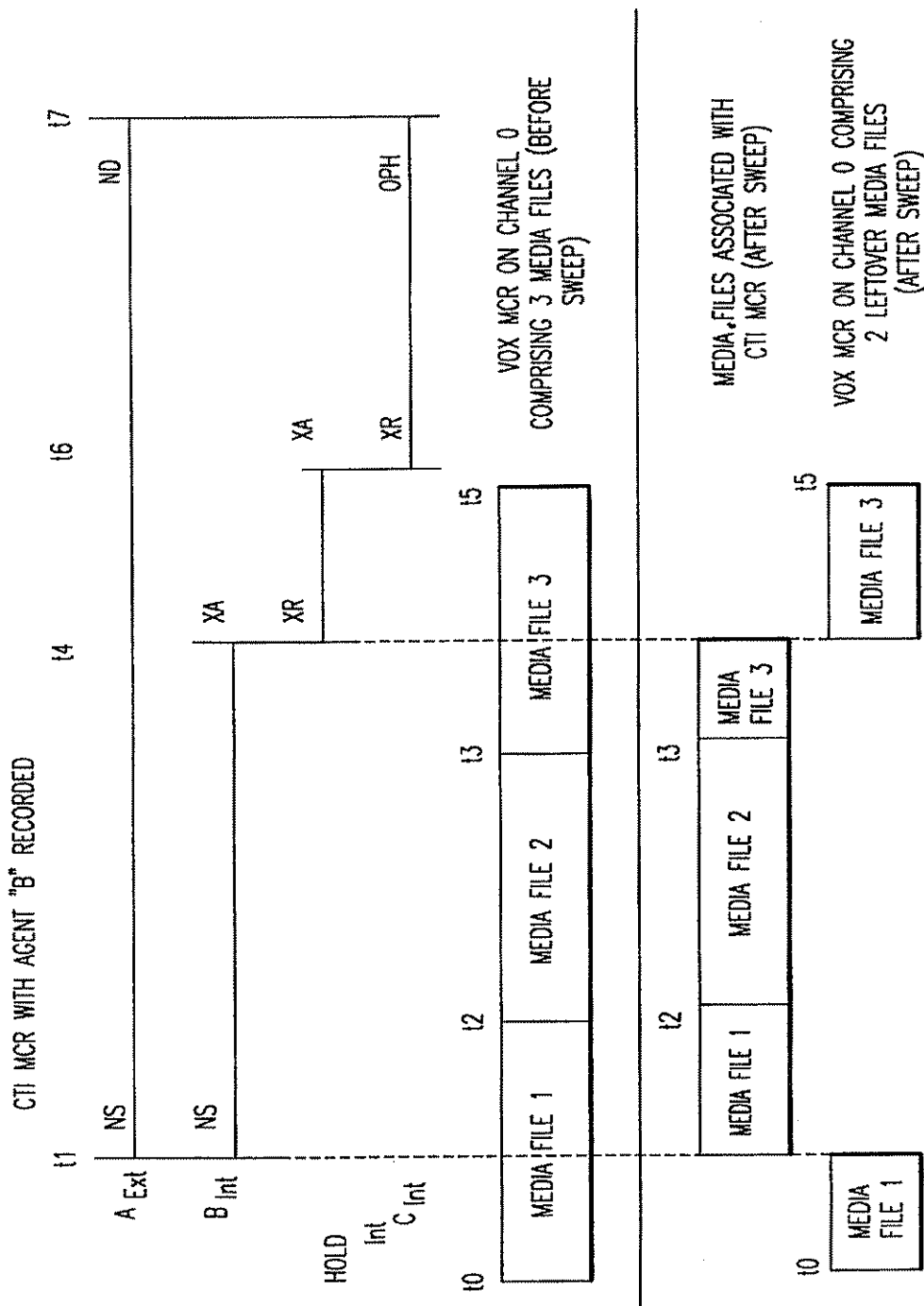


FIG.17

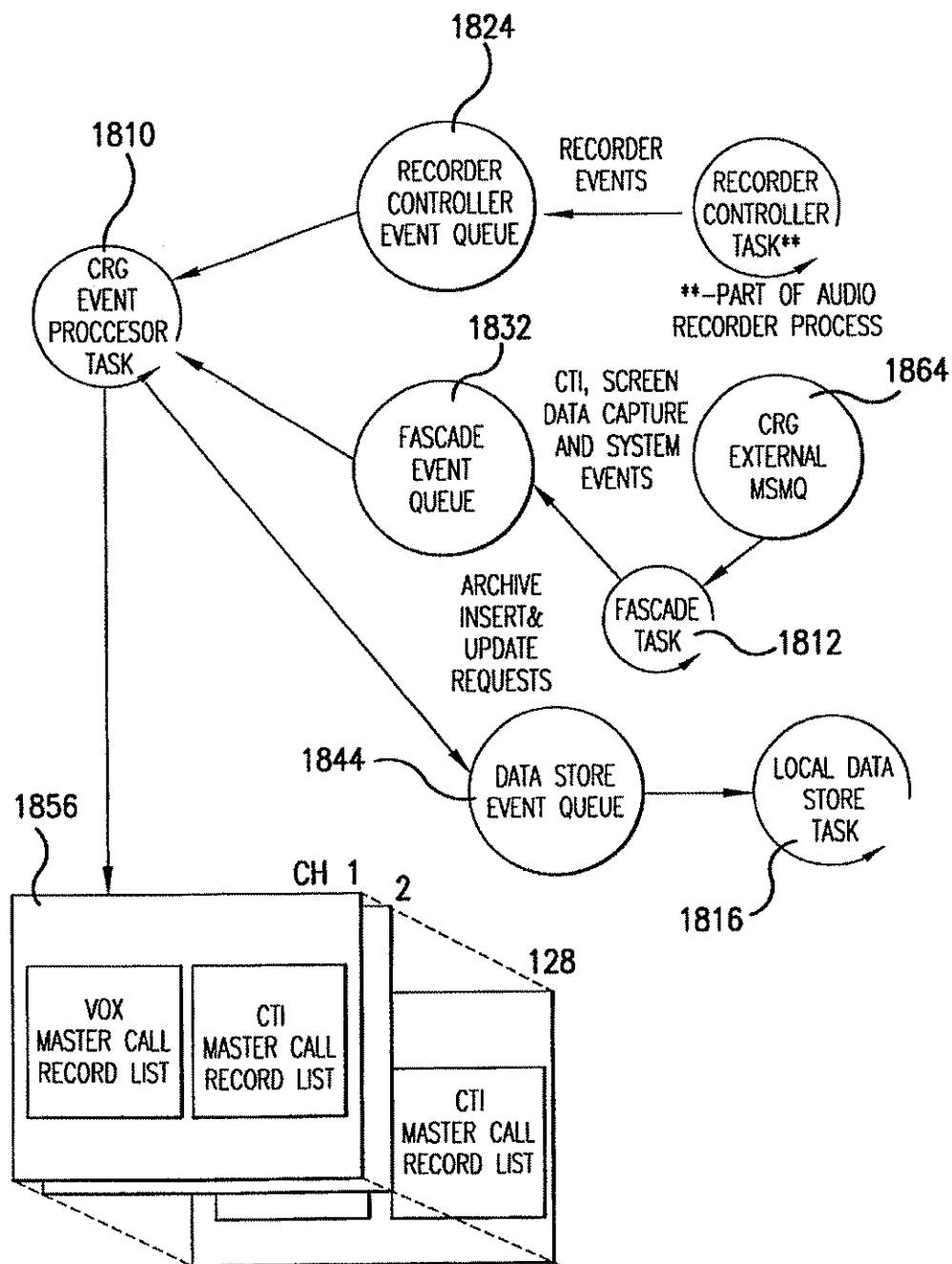


FIG. 18

U.S. Patent

Jun. 19, 2001

Sheet 21 of 29

US 6,249,570 B1

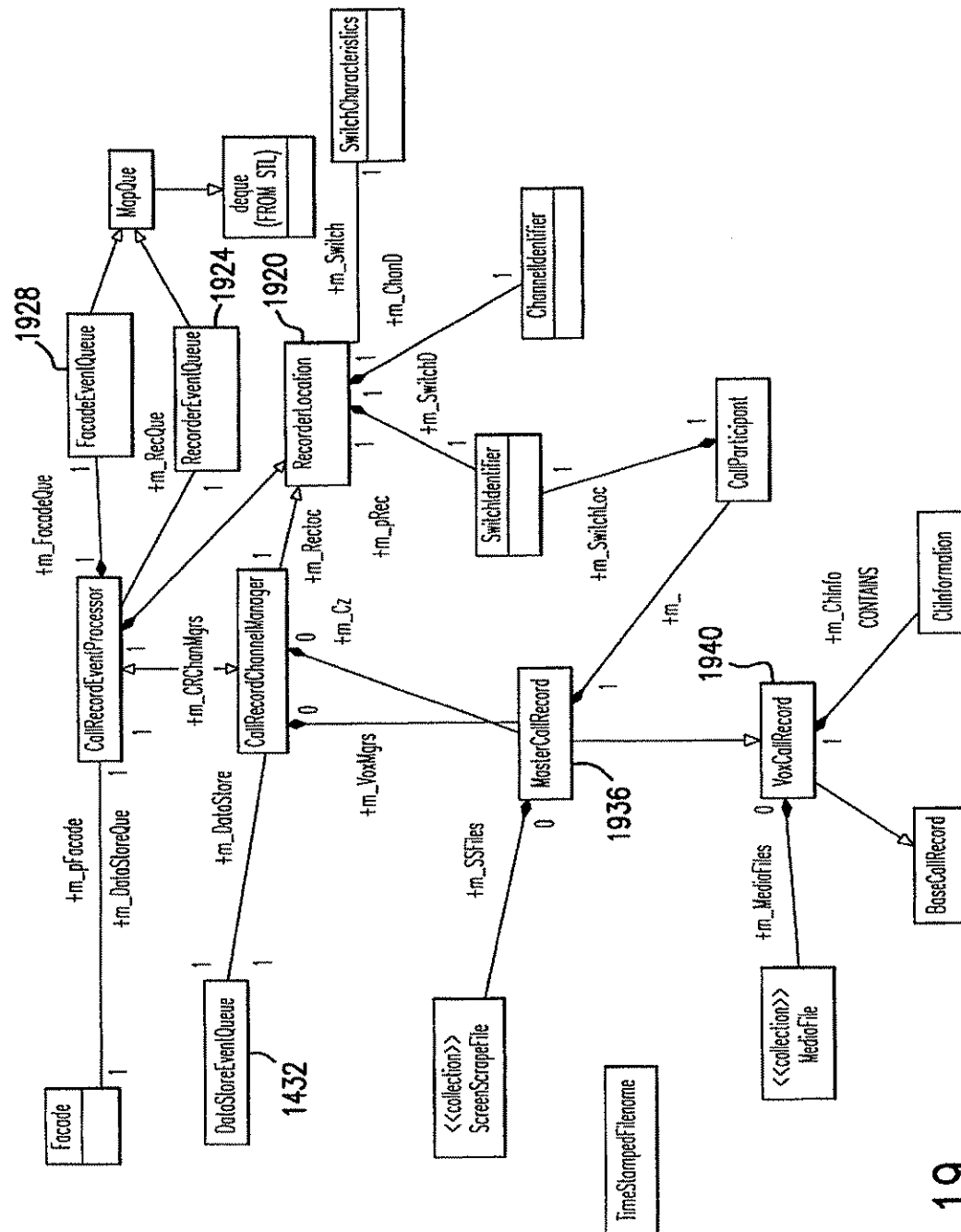


FIG. 19

U.S. Patent

Jun. 19, 2001

Sheet 22 of 29

US 6,249,570 B1

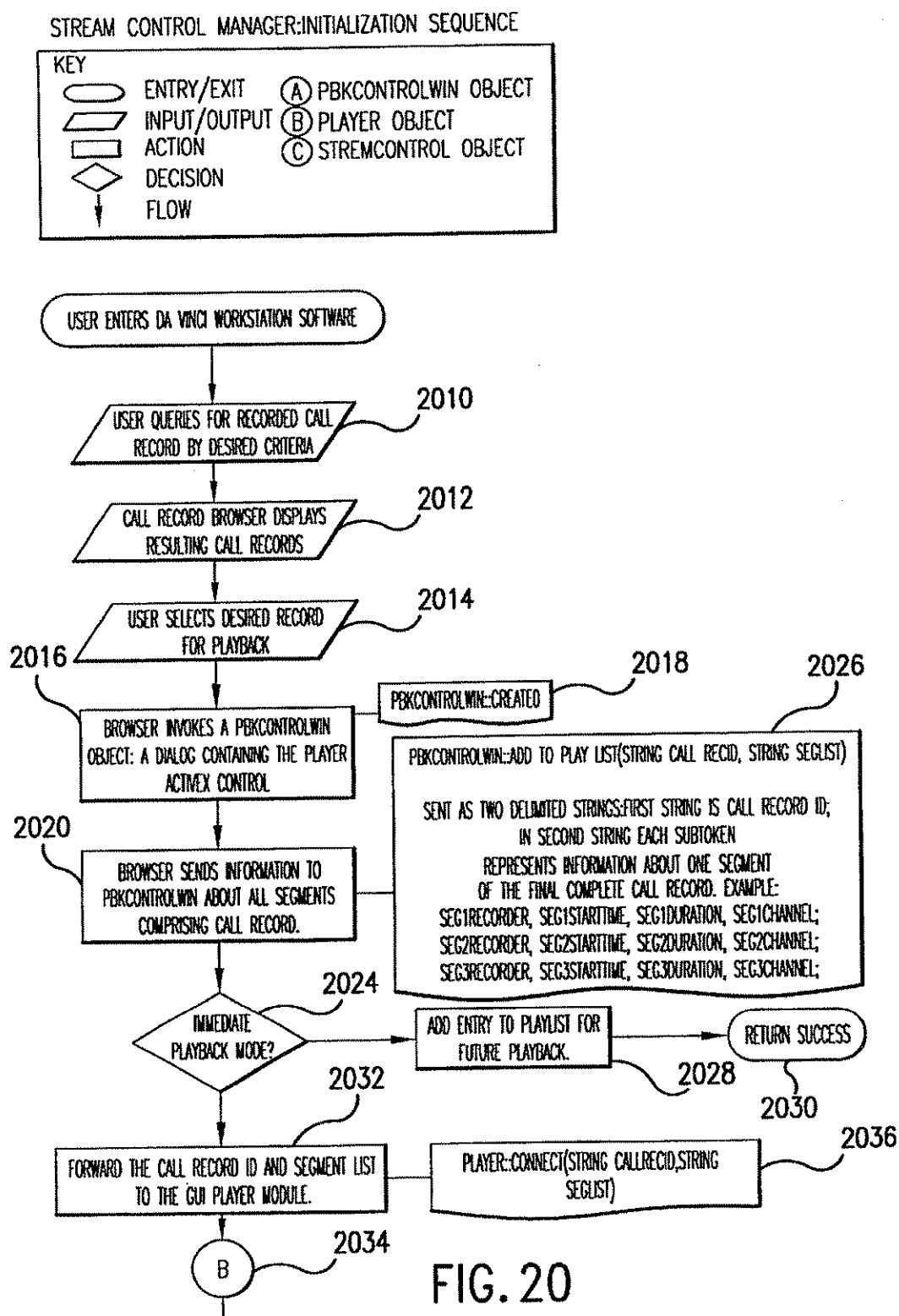


FIG. 20

U.S. Patent

Jun. 19, 2001

Sheet 23 of 29

US 6,249,570 B1

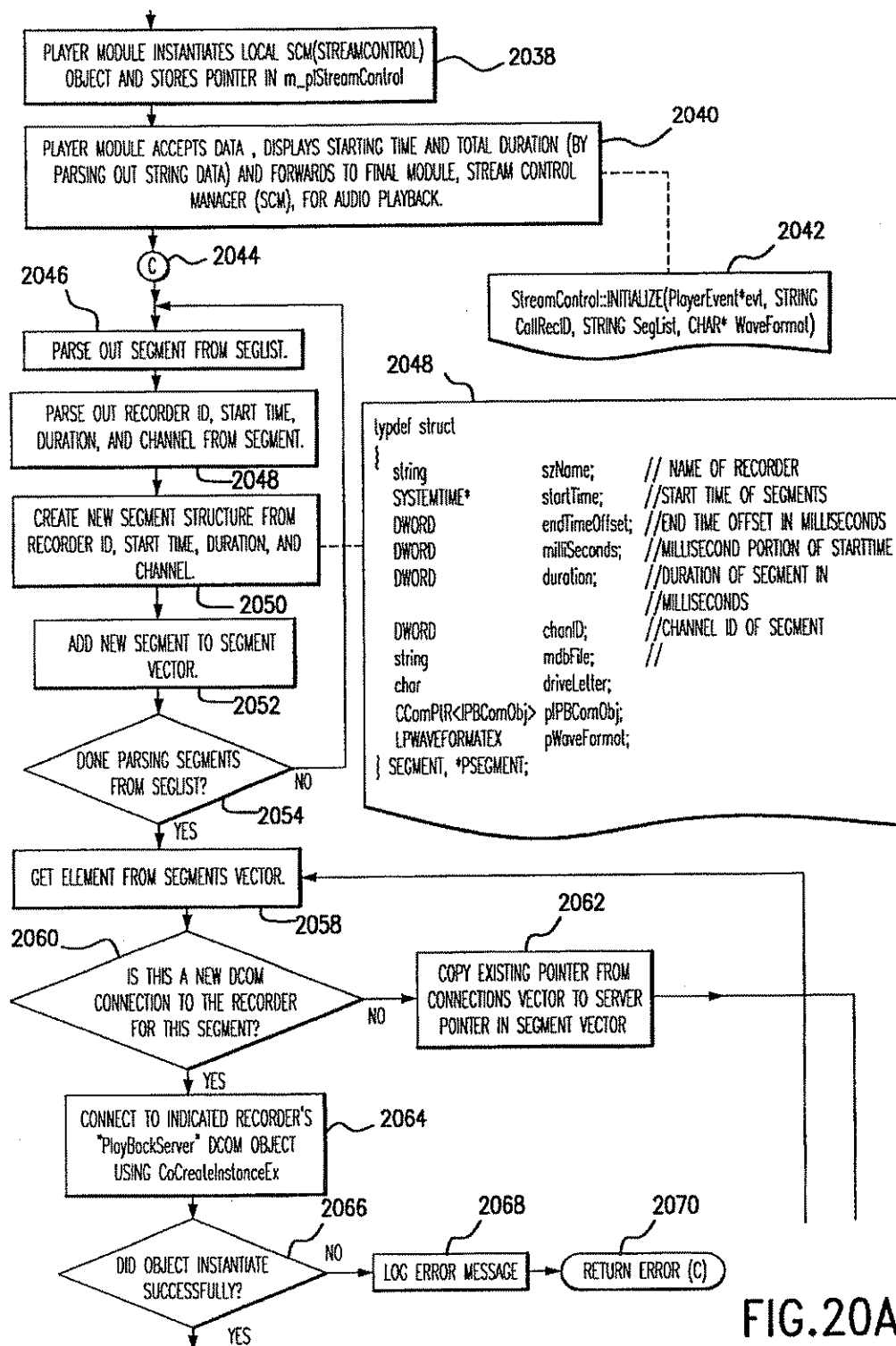


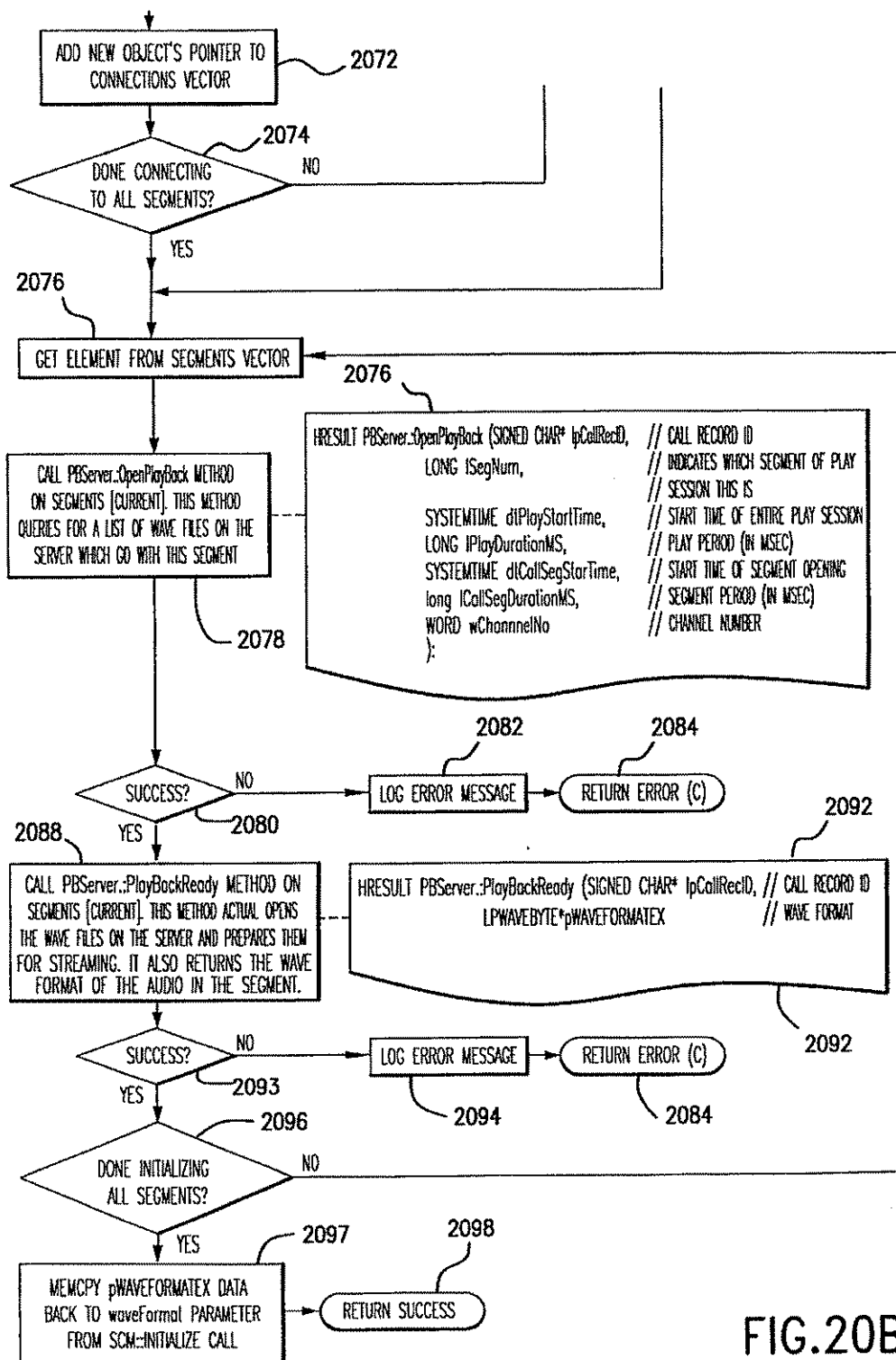
FIG. 20A

U.S. Patent

Jun. 19, 2001

Sheet 24 of 29

US 6,249,570 B1



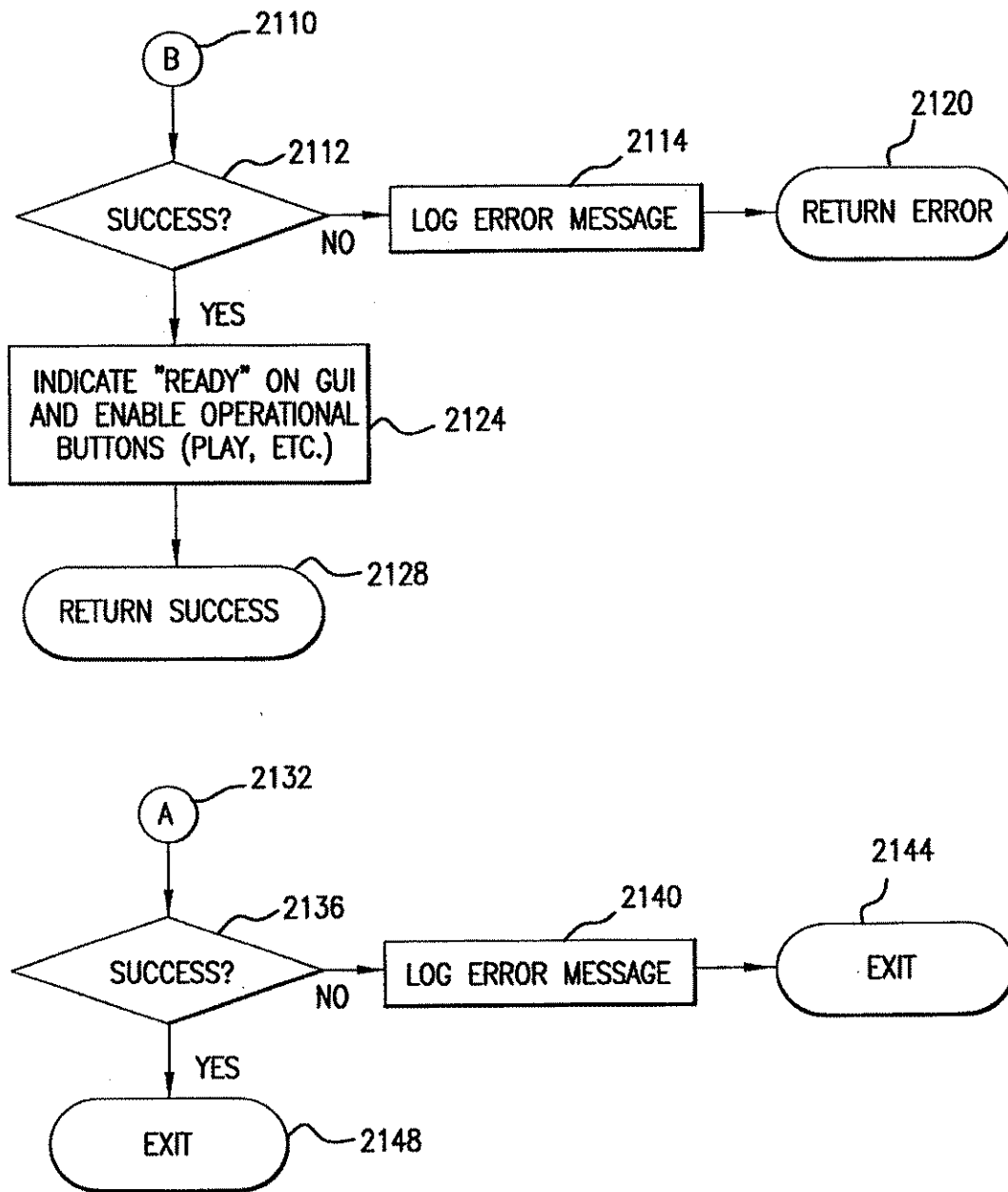


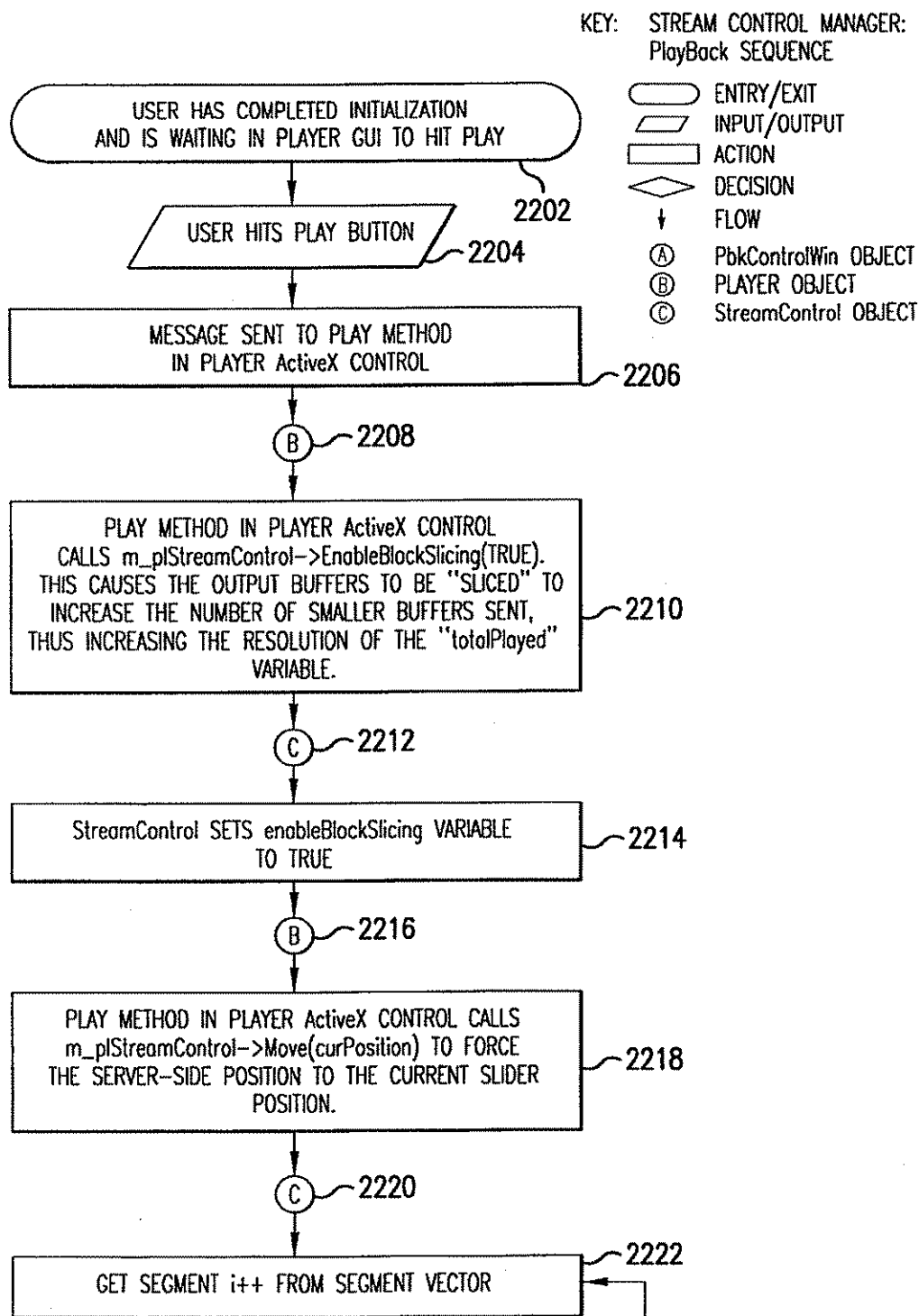
FIG.21

U.S. Patent

Jun. 19, 2001

Sheet 26 of 29

US 6,249,570 B1



U.S. Patent

Jun. 19, 2001

Sheet 27 of 29

US 6,249,570 B1

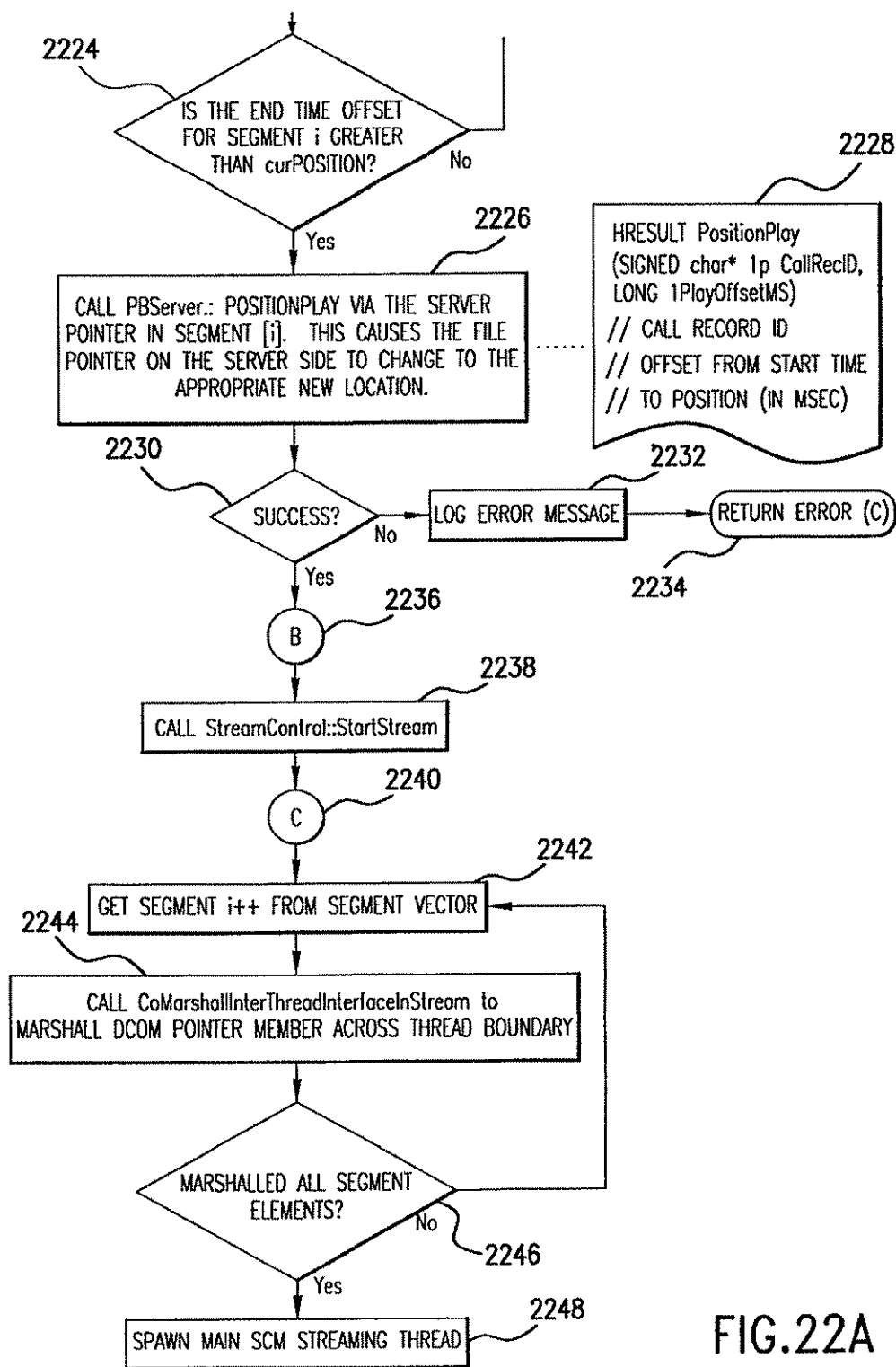


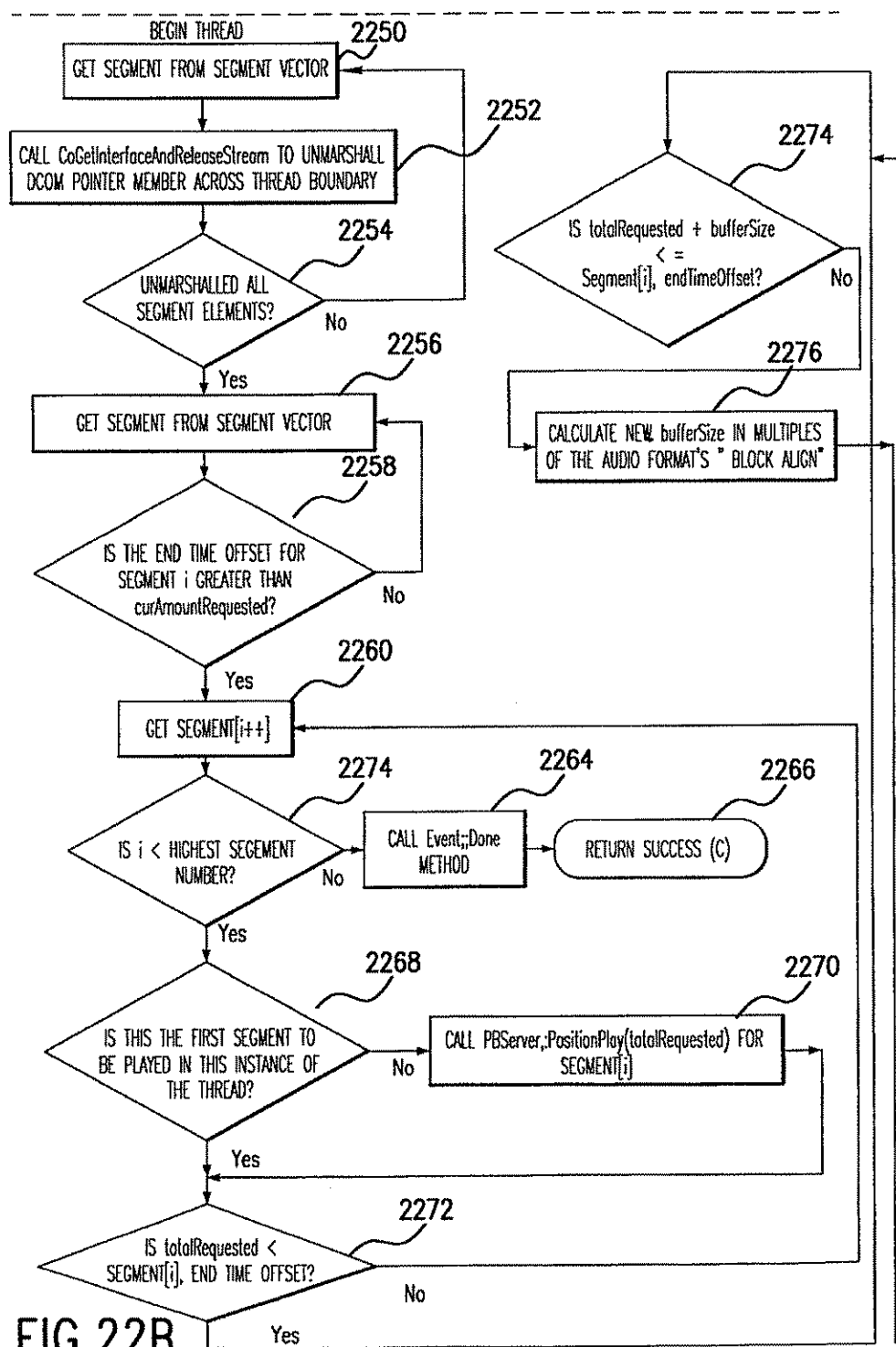
FIG. 22A

U.S. Patent

Jun. 19, 2001

Sheet 28 of 29

US 6,249,570 B1



U.S. Patent

Jun. 19, 2001

Sheet 29 of 29

US 6,249,570 B1

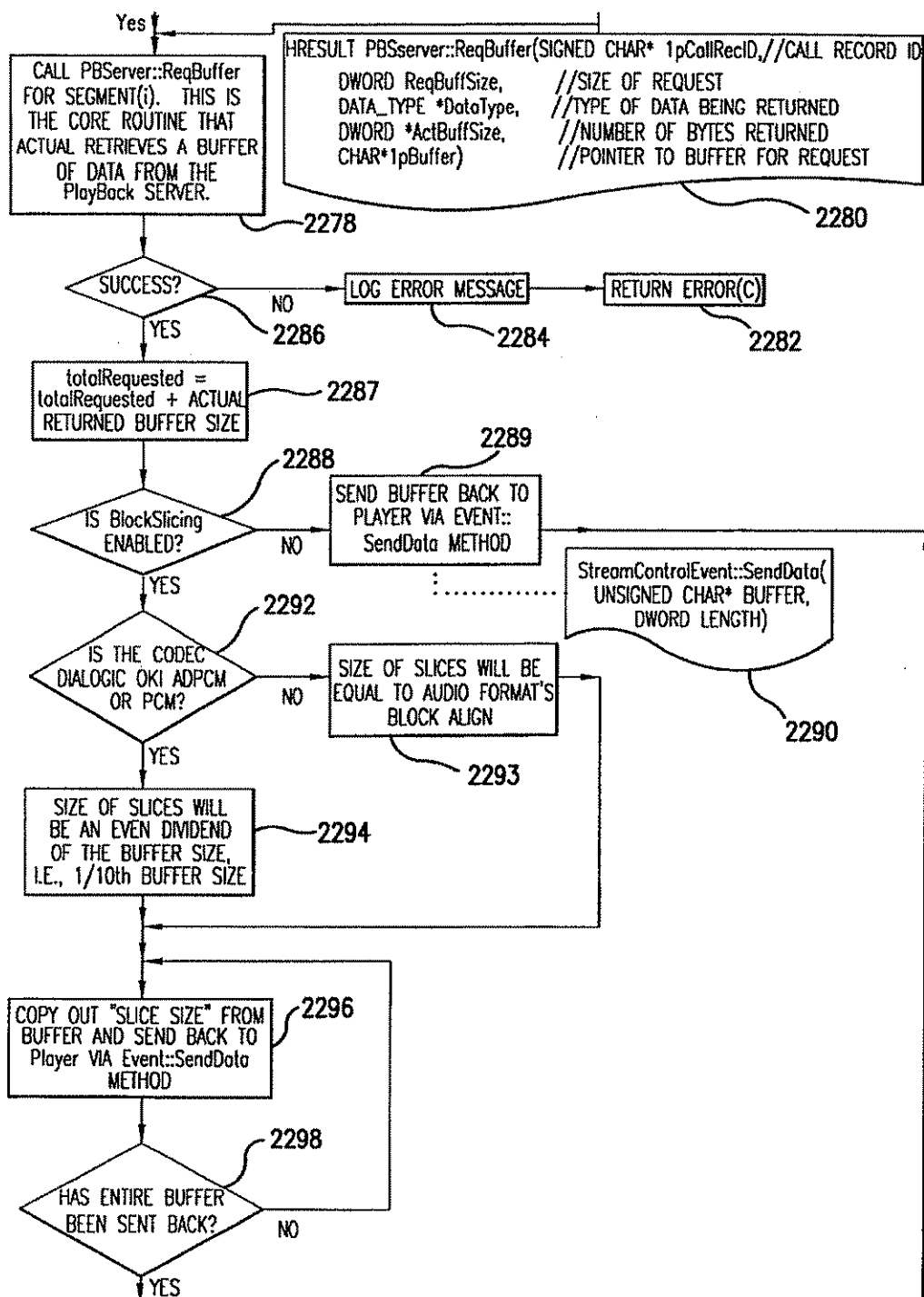


FIG.22C

US 6,249,570 B1

1

SYSTEM AND METHOD FOR RECORDING AND STORING TELEPHONE CALL INFORMATION

FIELD OF THE INVENTION

This invention relates generally to computer-aided data recording. In particular, it relates to computer-aided monitoring and recording of telephone calls.

BACKGROUND OF THE INVENTION

Telephone call monitoring systems are used in a variety of contexts, including emergency dispatch centers and commercial call centers. In many currently available call monitoring systems, a multitude of audio input sources ("channels") are monitored and recorded by a single hardware unit, and the audio recordings are saved and organized according to the input channel, date, time and duration. The capacity of the recording unit can be expanded to handle a larger number of channels by combining several recording units into a system using a local area network (LAN). Because retrieval is only possible using basic search criteria (recording unit, channel, date, time and duration), it is often difficult to locate a particular audio recording that is of interest. When there is a need to search for a recording according to search criteria that are not directly supported by simple voice recording, locating a specific recording may require tedious and repetitive searching. For example, if there is a need to find a specific customer's call to resolve a disputed transaction, the recording unit or channel that carried the original call might not be known, so the searcher would be forced to manually play back many calls before finding the correct one.

With the advent of computer telephony integration (CTI), it is now possible to monitor a data link that supplies more information about telephone calls, in addition to simple voice recording. In a typical CTI system a telephone switch or private branch exchange (PBX) provides an interface suitable for processing by a computer, and expanded information about telephone calls is made available through this interface as the calls occur. Data fields that are available within this expanded information may include the external telephone number of the calling party, as well as identification numbers to help associate a series of events pertaining to the same call. With such a data link being used alongside a voice recording system, the search and retrieval system can be supplemented by constructing a database that combines the previously discussed basic search criteria with enhanced search criteria (based upon information obtained through a CTI data link) such as: telephone numbers of parties involved in the call; Caller ID (CLID) or Automatic Number Identification (ANI); Dialed Number Identification Service (DNIS); or the Agent ID Number of the Customer Service Representative.

As shown in FIG. 2, with suitable equipment for tapping into a voice communications line, a recording unit can intercept telephone call traffic using two methods. By attaching wires for recording channels on each extension within a call center, the traffic can be intercepted and recorded as it passes between the PBX and the agent telephone set. This first method is known as "station-side" recording 180. Alternatively, by attaching equipment on the trunk lines between the PBX and Public Switched Telephone Network (PSTN), the traffic can be intercepted at its point of entry into the call center before the calls are dispatched by the PBX. This second method is known as "trunk-side" recording 170. Since businesses usually have more agent telephone

2

sets than trunk lines, a "trunk-side" solution is likely to require less recording equipment and thus be less expensive. Another significant point for consideration is that "trunk-side" provides access only to external inbound or outbound calls, which are those typically involving customers of a business, whereas "station-side" also provides access to internal calls between agents (which may or may not relate to an external customer's transaction).

With respect to data links to provide call information to computers, there are typically two different categories of links from the PBX available. Some older links use interfaces such as SMDR (Station Message Detail Recording) or CDR (Call Detail Recording) that provide summary information about telephone calls in a line-oriented text format. Both acronyms refer to essentially the same type of system. Information from these links is generally provided after the call has concluded, and as such is suitable for billing applications or traffic analysis software. Many newer links use real-time interfaces that are designed to supply a series of events while a telephone call is still active within the PBX, to enable computer and multimedia systems to respond and interact with an external caller. The information provided by such real-time links is typically much more detailed than that provided by SMDR.

The detailed information and real-time nature of a CTI link is particularly important when building a recording system that is intended to react to telephone calls as they occur and to dynamically select which calls ought to be recorded or discarded. CTI-supplied information is also important when building a recording system that is intended to capture the full history of a telephone call, including recording the different agents who were involved in the conversation and how the call was held, transferred or conferenced. Likewise, real-time information is important in a system that intends to support (a) a live display of active calls, and (b) the capability for a user to listen and monitor the live audio traffic.

A "trunk-side" solution based upon voice recording alone will not satisfy the above requirements in a practical manner, since telephone calls are assigned to trunks dynamically as needed to handle the traffic. What trunk channel a particular call will be carried on cannot be predicted in advance. Without information to associate a logical telephone call with a physical recording of audio from a trunk channel, a user might have to search and retrieve many recordings before finding the one that is of interest. Moreover, in a system designed to make use of the enhanced search criteria provided by a data link, it would not be possible to programmatically associate the search data with the voice recording without information about the trunk channel where the call occurred.

This problem can be avoided as long as the data link provides sufficient information about the trunk channels being used for each call. Unfortunately, some PBX environments do not supply this critical information about trunk channels within the data provided on the real-time CTI link. For example, this problem is manifested by the Lucent Technologies DEFINITY G3 PBX, which is a commonly used telephone switch in North America. While the Lucent G3 PBX provides trunk channel information through its SMDR link, that information is not available until after the conclusion of the call. This presents a problem for system features and capabilities dependent upon real-time data. The real-time data link provided by the Lucent G3 PBX does not provide the necessary information about trunk channels. There is thus a need for a system which is capable of simultaneously monitoring both the SMDR link and the

US 6,249,570 B1

3

real-time CTI link, gathering information about calls from both sources, and combining that information into a single data model of the telephony activity within the call center. There is a further need for a system that combines the data model with information concerning the location of call recordings, resulting in a "master call record" that contains data matching each call with the segments of which it is comprised, and matching the data for each segment with the location of the recording of that segment. Such a system would facilitate monitoring, recording, and playing back complete telephone calls.

SUMMARY OF THE INVENTION

The present invention is directed to a system and method that is capable of simultaneously monitoring two or more data links, gathering information about calls from those data links, combining that information into a single data model of the telephony activity within the call center, and combining the data model with information concerning the location of call recordings, resulting in a "master call record" that contains data matching each call with the segments of which it is comprised, and matching the data for each segment with the location of the recording of that segment.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of the system of this invention in a preferred embodiment.

FIG. 2 illustrates the difference between trunk-side and station-side recording.

FIG. 3 shows a line-chart that illustrates various parties involved in a complex call.

FIG. 4 shows a schematic block diagram of a preferred embodiment for translating, summarizing and normalizing signals received from both an SMDR link and a Dialogic CT-Connect CTI service.

FIG. 5 illustrates the steps by which the translation module CtiCtc.exe integrates the data received from the CTI and SMDR links.

FIG. 6 illustrates how the CTI Server can be viewed as a set of logically distinct layers that deal with translating and distributing CTI events.

FIG. 7 illustrates how, in addition to telephony events, the CTI Server 710 is responsible for supplying certain metadata regarding agent events to the System Controller 130.

FIG. 8 shows the layout of the CTI Server.

FIG. 9 shows a version of CtiCtc.exe configured to work with a Lucent Telephony Services interface (and thus called CtiLts.exe instead of CtiCtc.exe).

FIG. 10 depicts key elements of the data model used in a preferred embodiment.

FIG. 11 illustrates three distinct layers of the CTI Server in a preferred embodiment.

FIG. 12 shows in block diagram form several threads of the CTI Server in a preferred embodiment that implement three distinct layers of processing (data collection, data normalization, and message emission).

FIG. 13 illustrates the program logic flow of the analyzer layer of the preferred embodiment.

FIG. 14 depicts the flow of information within the recording system of this invention in a preferred embodiment.

FIG. 15 shows how a recording unit operating with only voice signaling to guide the creation of its call records could make a number of fragmented audio segments.

FIG. 16 shows a graphical user interface used in the preferred embodiment.

4

FIG. 16A shows a system containing a CTI Server and a Recorder in a specific embodiment of the present invention.

FIG. 16B is a table illustrating descriptive information from the CTI Server used in a specific embodiment.

FIG. 17 illustrates steps in the creation of a Master Call Record used in a specific embodiment.

FIG. 18 shows the processing threads and data structures that comprise the CRG module in accordance with the present invention.

FIG. 19 illustrates the class diagram of the Call Record Generator used in a specific embodiment.

FIGS. 20, 20A, 20B, 21, 22, 22A, 22B, and 22C illustrate the operation of the Stream Control Manager.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The present invention is directed to a communication recording system and method. Generally, the functionality of the system involves tapping into activity on a PBX (Private Branch Exchange) by intercepting audio on either the trunk or station side of a telephone call. The tapped audio is then redirected as input to a channel on a Digital Signal Processor (DSP) based voice processing board, which in turn is digitized into program addressable buffers. The recorded digitized audio is then combined with descriptive information ("metadata") obtained through a Computer Telephony Integration (CTI) communications link with the PBX, and stored as a single manageable unit ("Voicedata") to facilitate its subsequent search and retrieval. The system uses modular architecture in both its hardware and software, so that any one component can be replaced or upgraded without affecting the rest of the system.

In a preferred embodiment the communications recording system comprises multiple rack-mountable computer-processing servers (such as the Compaq ProLiant 1600 R), using a multi-tasking operating system (e.g., Microsoft Windows NT), DSP voice processing boards (e.g., Dialogic D/160SC), and a distributed set of software components available from Dictaphone Corporation. In a specific embodiment directed to the smallest configuration, all of these components may reside in a single computer-processing server. In other preferred embodiments, related components are typically packaged in combinations and the entire system spans multiple servers that coordinate processing through a Local Area Network (LAN).

In this preferred configuration, the overall system generally comprises CTI Servers, Voice Servers, a Central Database Server, and User Workstations. CTI servers generally use a set of components to manage a data communications link with a telephone switch environment, to obtain notification of calls as they occur, along with the descriptive information about the calls (e.g., source and destination telephone numbers). The Voice Servers use a set of components to collect audio recordings, manage their storage, and facilitate their playback through the LAN. The Central Database Server uses a set of components to manage system-wide search and retrieval of recorded calls. User Workstations are typically desktop computers that use a set of components to allow a person to submit requests to search and retrieve recorded calls for playback and to control automatically scheduled functions within the recording system.

FIG. 1 shows in a block diagram from components of the system of this invention in a preferred embodiment. Data enters the recording system from a variety of sources. These

US 6,249,570 B1

5

sources can include a PBX 100, CTI middleware 105, ISDN lines 110, or other input sources 115. It will thus be appreciated that the system of the present invention can be used for monitoring and recording of information about any type of electronic communications. For simplicity, the following discussion uses the term Telephone calls. However, it is intended that term covers any electronic communication unless specified otherwise expressly.

Data from data sources 100, 105, 110 or 115 is transmitted to one or more CTI Translation Modules 165, which translates input data into a common format. The data is then sent to a CTI Message Router 120, which distributes the data onward to appropriate components of the system.

Audio Recorders 145 may be used for passive trunk-side 170 and extension-side 180 recording on a predetermined static set of devices, as well as dynamically initiated recording of specific devices according to scheduling criteria through the Service Observance feature 185 provided by a telephone switch environment. The recordings are stored on an audio storage device 140. A Call Record Generator 150 matches data from the Audio Recorders 145 with data sent by the CTI Message Router 120 to create a Master Call Record (MCR) for each telephone call. The MCRs are stored in a Voicedata Storage module 155. One or more User Workstations 160 use the MCRs to reconstruct and play back complete or partial phone conversations stored in the audio storage device 140. A Scheduling and Control Services module 130 controls the Audio Recorders 145 and communicates with User Workstation 160. The Scheduling and Control Services module is responsible for starting and stopping the audio recording activity, according to pre-defined rules that are dependent upon time data provided by the Time Service 115 and CTI information. As the system components are packaged in the typical configuration, the CTI translation modules 165 and CTI message router 120 are co-resident upon a computer-processing server called the CTI Server 710. In a similar fashion, the combined set of components including the Time Service 125, Scheduling & Control Services 130, Audio Recorder 145, Audio Storage 140, and Call Record Generator 150, in a specific embodiment can be co-resident upon a computer-processing server called the Voice Server 124. The Voicedata storage 155 resides within a computer-processing server called the Central Database Server. The specialized application software for the User Workstation 160 resides upon desktop computers that use, in a preferred embodiment, Microsoft Windows 95, Windows 98 or Windows NT operating systems.

As noted above, in a specific embodiment the CTI Server comprises two main modules: a CTI translation module (such as the software program CtiCtc.exe, CtiLts.exe, and other translation modules) and a CTI Message Router module (such as the software program CtiServ.exe discussed below, or its equivalent). In a specific embodiment, the CTI Server may have several translation modules, for example, one for each PBX interface, or for each vendor API layer. As shown in FIG. 1, the CTI Server of the preferred embodiment accepts data from a PBX or similar equipment in a telephone switch environment, and can use both real-time CTI communications links and asynchronous information sources such as the Station Message Detail Recording (SMDR) interface. The CTI Server translates and combines the various types of input data into a unified, normalized format. Normalized format information is then passed by the Message Router to various components of the system, as required.

As noted above, the Voice Server in a specific embodiment has several modules, including the Audio Recorder

6

145 and Call Record Generator (CRG) 150. The Audio Recorder collects a plurality of audio segments, representing the portions of a telephone call during which the sound level exceeded an adjustable tolerance threshold, thereby discerning alternating periods of speech and silence. Functionally, the Call Record Generator (CRG) produces Master Call Records, which encapsulate information (metadata) describing a telephone call. This descriptive information comes from a plurality of sources, including but not limited to an Audio Recorder and a CTI Server. The call records are created using a participant-oriented Call Record Model. The CRG then attempts to match the call records with existing recorded audio data. The CRG is thus able to combine data arriving in different chronological order into a single manageable entity which describes the complete history of a telephone call.

In a specific embodiment, a Playback Server (PBServer) (not shown) is a sub-component within the Audio Recorder module which uses call records to retrieve and play back telephone calls. Each recorder has its own PBServer, which is connected to a Player module (not shown) on the User Workstation 160. The Player module generally contains a Stream Control Manager module, which enables the Player module to use the PBServers to play back a telephone call which has several different participants and thus may have portions of the call stored on different recorders.

CTI Server

Still with reference to FIG. 1, when a call comes into the PBX system, both SMDR and real-time CTI data are generated by the PBX, and supplied to the recording system via the SMDR and CTI links. In accordance with the present invention, these two types of data are integrated by the CTI Server into a common format.

As known in the art, CTI (Computer Telephony Integration) supplements the recorded audio data in several important ways. CTI data is provided through a data communications link from specific telephone switching equipment located at the customer site. Supplied data comprises such items as the telephone numbers of involved parties, caller ID/ANI information, DNIS information, and agent ID numbers. ANI is Automatic Number Identification, a signaling method that identifies the telephone number of the calling party; the method is typically used by large-scale commercial call centers. DNIS is Dialed Number Identification Service, a feature that identifies the original "dialed digits," and that is commonly used in large-scale commercial call centers when multiple directory numbers are routed to the same receiving trunk group. In accordance with the present invention, the CTI server performs the task of analyzing and reorganizing data from both the real-time (CTI) and SMDR (asynchronous) links, and passing the results onwards into the recording system for further processing.

The design of the system of the preferred embodiment envisions that there will be a number of CTI translation modules 165 to accommodate a variety of possible input sources such as "native" PBX interfaces, CTI "middleware" vendors, ISDN data channel interfaces, etc. The system design incorporates flexibility in the manner in which CTI information is collected, making the system prepared to integrate with CTI links that may already exist at a customer site. The CTI Server of the preferred embodiment is capable of simultaneously monitoring both an SMDR link and a real-time CTI link, gathering information about calls from both sources, and combining that information into a single data model of the telephony activity within the call center.

US 6,249,570 B1

7

The CTI Server is responsible for supplying certain metadata regarding telephony events to the Voice Server's Call Record Generator 150. This metadata, such as called party and calling party numbers, trunk and channel ID, date and time, agent ID, etc., is combined by the Call Record Generator along with the other metadata, and data that is provided by the Audio Recorder 145 itself. Using this information, other components within the system are able to search for calls using a wide variety of useful and meaningful criteria, rather than simply using the recorder channel, date and time. As is known to those skilled in the art, an "event" is simply an action or occurrence detected by a computer program. The Call Record Generator 150 integrates that data into a single call record, which is updated after every event during the call, so that at the end of the call, the call's entire history is contained in the call record. The CRG matches the call record to the recording segments created by the Audio Recorders. The CRG integrates the call record with the metadata for the associated recordings of the phone call to generate a Master Call Record. When an operator wants to hear a recorded phone call, he uses the User Workstation (preferably equipped with a graphical user interface) to recall and play back the recorded call. Since the phone call may have had several different participants, pieces of the call may have been recorded on different recorders, each of which is associated with a different Playback Server. The system is nevertheless capable of playing back the entire phone call in the proper sequence.

In a preferred embodiment the CTI Server obtains the information regarding telephony events from various telephone switching environments, including PBXs, ACDs, and turret systems, which may have a wide variety of proprietary CTI interfaces. A telephone switching environment is a local telephone system that provides for routing of calls on a static or dynamic basis between specific destinations; the system is capable of identifying of when calls occur and who is involved in the calls. The CTI Server converts the information received into a common "normalized" format that is a simplified subset of the types of information available across the different vendors' PBXs, ACDs, and turret systems. This data conversion is partially facilitated by products such as Dialogic's CT-Connect API, which is capable of processing CTI messages from the major vendor's switches such as the Lucent DEFINITY G3, Nortel Meridian and DMS-100, Aspect, Rolm 9751, Rockwell Spectrum and Galaxy, Siemens Hicom, and Intecom. However, in accordance with the preferred embodiment an additional software layer exists within the CTI Server to further filter and normalize the CTI information. This feature also allows for a separate point of integration with customized software interfaces that may be necessary to connect with other switch vendors, especially certain turret systems that are not supported by Dialogic's CT-Connect (CTC) product. Alternate embodiments of the translation module use Lucent CentreVu Computer Telephony Server for Windows NT, or Genesys T-Server, as middleware instead of Dialogic CT-Connect. Additional alternate embodiments include direct "native" interfaces to a particular telephone switch, such as Aspect, without an intervening middleware product.

In terms of the CTI messages exchanged between the CTI Server and the various PBXs, ACDs, and turret systems, in accordance with a preferred embodiment the CTI Server is a "passive listener." That is, the CTI Server will monitor and receive information about call activity, but it will not send messages to affect, control, or redirect the calls. Using an "active" CTI server is also contemplated in specific embodiments.

8

Whereas the focal point of a Voice Server is recording content (e.g., audio clips), the metadata generated by the CTI Server is focused on describing the facts pertinent to the start and end points of each participant's involvement within a call. In other words, within the system of the preferred embodiment, recording is managed in a call-centric (rather than event-centric) fashion. This corresponds with the typical caller's point of view, in which a call is the entire conversation with a business entity, even if the conversation involves transfers to other agents or conferencing of multiple parties. The CTI Server generates events with metadata for the start and end points of the various recording segments of a complex conversation. These event records are interrelated by ID numbers and reason codes (see FIG. 3) so that the entire sequence of events for a complex conversation can be reconstructed by a browser application, preferably implemented on the User Workstation 160.

In accordance with the preferred embodiment, there can be one or more CTI Servers within the system of the subject system, as needed to process the traffic load of CTI information generated by multiple PBXs, ACDs, and turret systems. In a specific embodiment, a single CTI Server may be configured so as to connect with several PBXs, ACDs, and turret systems, depending upon the traffic load and physical connectivity requirements. In alternate embodiments, different CTI servers can be attached to different input sources. Generally, the number of CTI Servers within the system does not have a direct relationship with the number of Voice Servers. The telephony events generated by a CTI Server are individually filtered and re-transmitted to the appropriate Voice Server based upon configuration data for the system as a whole (managed by the Central Database Server), which maps the recording locations (extension number, or trunk & channel ID) with the Voice Server name and recording input port (channel).

During the active lifetime of a call, real-time information is accumulated within a historical call record that tracks each participant within the call. Each participant record includes descriptive fields for telephone numbers, agent ID numbers, time ranges, and reason codes for joining and leaving the conversation. At certain key points during the accumulation of data, whenever a party joins or leaves the conversation, the call record is transmitted onward to allow the rest of the recording system to process the information accumulated thus far. Upon the conclusion of the call, the CTI server retains a copy of the call record for a configurable time interval before discarding it from memory. This delay is intended to allow for the arrival of the SMDR data.

Upon receiving SMDR data, the CTI server searches its memory for a call record pertaining to the same logical telephone call that would have been accumulated from previous real-time messages. Matching this information is not a trivial task, since the SMDR link and real-time CTI link do not share a common reference ID number for use by their messages in describing the occurrence of the telephone call.

Therefore the software of the preferred embodiment must use other "clues" to guide the matching process, by comparison on a combination of other data fields that exists in common between the SMDR and real-time CTI data. These data fields include: (1) the telephone number of the first external party involved in the call; (2) the telephone number of the first internal party involved in the call; (3) the direction of the call (e.g., inbound, outbound); (4) the start time of the call, in hours and minutes; and (5) the duration, in seconds, of the call.

Once again, the matching process is not trivial because the SMDR link gives the starting time of the call only in hours

US 6,249,570 B1

9

and minutes, whereas the starting time given by the real-time link also includes seconds. It is quite possible that more than one call could be started and stopped within a single minute. This would result in an ambiguous match, if not combined with other search fields. The same argument holds true for each of the other fields upon which a match can be performed. No single field alone will provide an unambiguous matching of the records. Even in combination, it is conceivable (although statistically unlikely) that an ambiguous case could occur: if the same two parties were to call each other twice within the span of a minute, and each call was roughly the same length in seconds. The odds of such a problem are increased if a large number of calls are routed through a common entry point into the call center, as would be the case if the first internal party involved in the call is a shared Voice Response Unit (VRU) or Automatic Call Distribution (ACD) queue. In addition, if information about the external party's number is missing due to limitations of the PSTN or incoming trunk configuration, matching the call records becomes even more problematic.

Adding to these difficulties is the fact that clock-time values reported by the SMDR link and the real-time CTI link may not be perfectly in synchronization with each other. Therefore, the preferred embodiment comprises a mechanism in which an imperfect match of times can be tolerated, while still retaining an acceptable level of reliability in matching the call records.

Because these various factors require a degree of flexibility in the matching algorithm, the preferred embodiment incorporates a weighted formula that is applied to potential match candidates. The formula yields a numerical confidence factor that can be used to select the best apparent match candidate. For each of the "clues," a test is conducted to determine the quality of matching on that data field. This matching quality is rated as a percentage. Certain fields, such as time values, are allowed to vary within a configurable tolerance range, whereas other fields are required to match exactly or not at all. After the matching quality of a field has been determined, it is multiplied by an importance factor that applies a relative weight to each of the various fields that can be examined during matching. The final confidence factor is the summation of these calculations:

$$\text{Confidence Factor} = \sum_i ((\text{Match Quality})_i * (\text{Weighting Factor})_i)$$

In order to account for the fact that characteristics of the call traffic may vary significantly between individual call centers, the tolerance factors (e.g., for time value offsets) and the weighting factors are re-configurable. There is also a re-configurable minimum level for confidence factors, below which the match candidate will always be rejected.

For those fields, such as time or duration, where an imprecise match may be allowed, the configuration data will define an allowable variance range (plus or minus a certain number of seconds). Values that do not match exactly, but fall within the variance range, are rated with match quality expressed in percentage that is measured by one minus the ratio of the difference from the expected value versus the maximum variance.

$$\text{Match Quality} = 1 - (\text{abs}(\text{Expected Value} - \text{Actual Value}) / \text{Maximum Variance})$$

Values outside the variance range are rated as a match quality of zero. This produces a linearly scaled match quality. Alternate embodiments may use other distributions (e.g., standard deviation "bell curves") to produce a non-linear scale for the match quality. Where an exact match is required for a field, the match quality is either 100% or zero.

10

Example Real-time CTI events report a telephone call from an unknown external party missing or deliberately suppressed ANI/CLID information) to an internal party at extension 1234, starting at 12:25:03 and lasting for 17 seconds (CLID is Calling Line Identification, a signaling method that identifies the telephone number of the calling party; the method is typically used by residential subscribers and small businesses). Two SMDR records arrive which could possibly match with this call. The first record indicates an inbound call received by extension 1234 at 12:26 and lasting 26 seconds. The second record indicates an inbound call received by extension 1234 at 12:27 and lasting 20 seconds. The system is configured with a variance range of plus or minus 3 minutes for the start time, and plus or minus 10 seconds for the duration.

Weighting Factors are:

- 20 External Party Telephone Number
- 40 Internal Party Telephone Number
- 30 Direction
- 20 Start Time
- 20 Duration

Confidence Factors are therefore calculated as follows:

$$CF_1 = (20 * 1.00) + (40 * 1.00) + (30 * 1.00) + (20 * (1 - \frac{1}{3})) + (20 * (1 - \frac{2}{10})) = 105 \frac{1}{2}$$

$$CF_2 = (20 * 1.00) + (40 * 1.00) + (30 * 1.00) + (20 * (1 - \frac{2}{3})) + (20 * (1 - \frac{3}{10})) = 110 \frac{3}{4}$$

The system will therefore match the CTI events with the second SMDR record.

After a match has been selected, the trunk channel information (and any other useful information that can supplement the previously gathered real-time CTI data) is extracted from the SMDR data and added to the call record within the CTI server's data model of telephony activity. Then the updated call record is transmitted onward to allow the rest of the recording system to process it. With the trunk channel information at hand, the recording system is able to associate the enhanced logical search information with the physical voice recording, and take whatever actions may have been dependent upon this information, such as selectively recording or discarding the call.

FIG. 2 is an illustration of the difference between trunk-side and station-side recording at a call center with agents.

With suitable equipment for tapping into a voice communications line, a recording unit can intercept telephone call traffic using either of these two methods. By attaching wires for recording channels 180 on each extension within a call center, the traffic can be intercepted and recorded as it passes between the PBX 100 and the agent telephone sets 230. This first method is known as "station-side" recording. Alternatively, by attaching equipment 170 on the trunk lines between the PBX and Public Switched Telephone Network (PSTN) 250, the traffic can be intercepted at its point of entry into the call center before the calls are dispatched by the PBX. This second method is known as "trunk-side" recording. Since businesses usually have more agent telephone sets than trunk lines, a "trunk-side" solution is likely to require less recording equipment and thus be less expensive.

Another significant point for consideration is that "trunk-side" provides access only to external inbound or outbound calls, which are those typically involving customers of a business, whereas "station-side" also provides access to internal calls between agents (which may or may not relate to an external customer's transaction).

A third type of recording interface is Service Observance 185 (see FIG. 1), which is physically wired in manner like

US 6,249,570 B1

11

station-side recording, but using separated dedicated lines to a recording input channel rather than being interposed between a PBX and telephone set. In this mode of operation, the Recorder joins into a telephone call as a silent conference participant using the PBX Service Observance feature (originally intended to enable a supervisor to directly monitor an employee's telephone calls upon demand). This differs from ordinary station-side recording in that the internal party being recorded on a given input channel can vary upon demand rather than being fixed by the wiring pattern.

FIG. 3 shows a line-chart that illustrates various parties involved in a complex call. A is the customer phone number, and B and C are the agent phone numbers located behind recording channels R20 and R21 respectively (see FIG. 2).

Initially, the call comes in from line A 335 to line B 340. A real-time CTI message occurs describing that phone B is ringing, but not yet answered. B answers the phone 365 at time t0 310. The "NS" at 360 indicates the normal start of a phone call. A real-time CTI message occurs describing the start of the call between A and B. The telephony model is updated to reflect the fact that the call between the initial 2 participants (A and B) started normally at time t0 310. A copy of the call record is then sent onward to the rest of the recording system. The call record is retained within the telephony model, associated with device (or line) B. At time t1 315, B places the call on hold 370 (the "XA" at 370 indicates that the call was transferred away from B; the "XR" at 375 indicates that the transfer was received by HOLD). A real time CTI message occurs describing that B placed the call on hold. The telephony model is updated to reflect that B transferred the call to HOLD 345 at time t1 315. (This information is accumulated with the information previously gathered at t0 310). A copy of the call record is then sent onward to the rest of the recording system. The call record is removed from device B within the telephony model, but kept in a list of held calls.

At time t2 320, B returns to the call 380 and conferences in C 355 (the "XA" at 380 indicates that the call was transferred away from HOLD; the "XR" at 382 indicates that the transfer was received by B; the "CA" at 384 indicates that C was added as a conference participant). A real-time CTI message occurs describing that B returned to the call and invited C by conferencing. The call record is moved within the telephony model from the list of held calls back to device B. The telephony model is updated to reflect that HOLD 345 transferred the call 380 back to B at t2 320. (Note that information is accumulated with the information previously gathered at t0 310 and t1 315). A copy of the call record is then sent onward to the rest of the recording system. The telephony model is updated to reflect that C joined the call 384 as a conference participant at t2. (This information continues to be accumulated with previously gathered information). A copy of the call record is then sent onward to the rest of the recording system. The call record is retained with both devices B and C within the telephony model.

At time t3 325, a real-time CTI message occurs describing that C dropped out 386 of the call (the "CD" at 386 indicates that C was dropped from the conference). The telephony model is updated to reflect that C dropped out of the conference at t3. (This information continues to be accumulated with previously gathered information). A copy of the call record is sent onward to the rest of the recording system. The call record is removed from device C within the telephony model, but retained with device B.

At time t4 330, A terminates the call to B. A real-time CTI message occurs describing that A terminated the call (The

12

"ND" at 390 indicates that a normal drop of the call occurred; the "OPH" at 395 indicates that the other party hung up). The telephony model is updated to reflect that A stopped normally and B stopped because the other party hung up at t4. (This information continues to be accumulated with previously gathered information). A copy of the call record is then sent onward to the rest of the recording system. The call record is then removed from device B, but kept in a list of completed calls. An SMDR message is received which summarizes the call in its entirety. The list of completed calls is searched to find a match, and the appropriate call record is retrieved. The call record is updated with the trunk channel information from the SMDR message. A copy of the call record is sent onward to the rest of the recording system. The call record is removed from the list of completed calls.

FIG. 4 shows a schematic block diagram of a preferred embodiment for translating, summarizing and normalizing signals received from both an SMDR link and a Dialogic CT-Connect CTI unit. In the embodiment illustrated in FIG. 4, the recording system of the subject system is represented by daVinci™, a new generation recording system of Dictaphone Corp. Alternatively (or simultaneously), Dictaphone's Symphony™ CTI software can be used, in conjunction with Dictaphone's ProLog™ recording system (the system preceding daVinci™). Hereinafter, the translation/summarization module of the preferred embodiment illustrated in FIG. 4 will be referred to as CtiCtc.exe.

The module CtiCtc.exe is itself comprised of a plurality of modules, as shown in FIG. 4. A CtiAgentEvent module 448 is comprised of a data structure for agent log-on and log-off messages. A CtiAgentStatusFile module 454 manages a file that tracks agents currently logged on. A CtiCallEvent module 416 is comprised of a data structure for a call record (i.e., normalized and summarized CTI events). A CtiCall-State module 418 is comprised of a generic data structure to represent the state of telephony activity at a particular location (extension, hold area, etc.). A CtiComMessageEmitter module 476 comprises a layer that converts the stream of CtiCallEvent objects (generated by a CtiCtcAnalyzer 456) into a format that can be sent to other da Vinci system components. A CtiCtcAnalyzer module 456 comprises a processing engine which examines CTC and SMDR messages and keeps track of a state machine for the activity on each extension. The CtiCtcAnalyzer module performs normalization of the CTC and SMDR data.

A CtiCtcAnalyzerUtils module 452 comprises a collection of utility subroutines that assist in examining the CTC and SMDR messages. A CtiCtcCallState module 420 comprises a data structure that represents the state of telephony activity at a particular location (extension, hold area, etc.) including CTC-specific information. A CtiCtcCallStateList module 432 manages an open-ended collection of CtiCtcCallState objects. This collection of objects is typically used to track calls that are "held" or "bumped." A CtiCtcData module 428 comprises a data structure wrapped around the raw CTC data, with the addition of a time stamp indicating when a message arrives. A CtiCtcDataFile module 412 manages a file of CtiCtcData objects that can be captured or displayed. A CtiCtcExtensionInfo module 442 manages a collection of CtiCtcCallState objects, with one object for each extension.

A CtiCtcInput module 464 comprises an input source engine that obtains incoming CtiCtcData objects, either from a "live" server or from a playback file. A CtiCtcMain module comprises the main() function for CtiCtc.exe. The main() function handles command line and registry

US 6,249,570 B1

13

parameters, along with other start-up processing. A CtiCtcParameters module 472 comprises data structure and program logic for managing the configuration parameters in the Windows NT registry. A CtiCtcScanner module 446 comprises a utility module for building a list of all available extensions on a particular telephone switch. A CtiCtcStats module 434 comprises a data structure for compiling statistics on the number of CTC, SMDR, and CTI messages. A CtiDtpField module (not shown) is used by a CtiDtpMessageEmitter module 478, and comprises a data structure for an individual field in the Dictaphone Telephony Protocol ("DTP"), used to communicate with other Symphony CTI system components. A CtiDtpMessage module (not shown) is used by a CtiDtpMessageEmitter module 478, and comprises a data structure for a complete message in the DTP to be sent onwards to the Symphony CTI system.

A CtiDtpMessageEmitter module 478 comprises a layer that converts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) into a format that can be sent to the Symphony CTI recording platform. A CtiDtpSocketSrv module (not shown) manages the TCP/IP connection through which messages for DTP are sent to the Symphony CTI platform. A CtiDtpUtility module (not shown) comprises a collection of utility routines that assist in examining and processing DTP messages. A CtiExtensionFile module 450 manages the configuration file that lists all available telephone extensions. A CtiExtensionInfo module 440 manages a collection of CtiCallState objects, with one object for each extension. A CtiExtensionNumber module 430 comprises an abstraction of an individual extension number as either a numerical or string value, so that changes to this model will not have a global impact in CtiCtc.exe.

A CtiMessageEmitter module 458 comprises an abstract layer that converts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) into a format that can be sent to various target platforms, including the da Vinci and SymphonyCTI systems. A CtiMessageEmitterParameters module 474 comprises a data structure and program logic for managing configuration parameters that relate only to the message emitter(s). A CtiMessageQueue module 462 comprises shared memory for transferring data between threads. As is known to those skilled in the art, a "thread" is a part of a program that can execute independently of other parts. A CtiNullMessageEmitter module 460 comprises a layer that accepts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) and discards them instead of sending them to a target platform. Typically this layer is used only when debugging CtiCtc.exe, or to capture a sample file of CTI events from a PBX without sending them to the da Vinci or SymphonyCTI systems. A CtiPartyListElement module 414 comprises a sub-component of the CtiCallEvent data structure 416. The module 414 tracks information about an individual participant (e.g., caller, recipient) in a call.

A CtiPeriodicMsg module 468 comprises a generic handler for sending timer-based housekeeping messages. A CtiPrint module 444 comprises a layer that manages console output and conditional trace messages. A CtiSmdrData module 424 comprises a data structure wrapped around the raw SMDR data, with the addition of a time stamp indicating when a message arrives. A CtiSmdrDataFile module 408 manages a file of CtiSmdrData objects that can be captured or replayed. A CtiSmdrDataList module 422 manages an open-ended collection of CtiSmdrData objects. This is typically used to buffer SMDR records that have not been paired with CTC records. A CtiSmdrInput module 466 comprises an input source engine that obtains incoming CtiSmdrData objects, either from a "live" server or from a playback file.

14

A CtiTagNames module 436 comprises a utility module that converts number values to descriptive strings for debugging and tracing purposes. A CtiTime module 438 comprises a utility module that converts time values to UTC for internal storage and conditionally prints times in either the UTC or local time zone. A CtiTrunkMap module 426 comprises a data structure that describes a mapping between logical trunks and logical trunk groups, into physical trunks and TDM timeslots. A CtiTrunkMapFile module 410 manages a configuration file that contains the CtiTrunkMap information.

FIG. 5 illustrates the steps by which the translation module CtiCtc.exe integrates the data received from the CTI and SMDR links. Initially, at step 502, the translation module receives a message from the SMDR link or from the CTI link. If the message is determined, at step 504, to be a CTI message, the current data model of telephony activity is updated at step 506. If the translation module determines at step 514 that the CTI message indicates a party joined or left the call, the call record is at step 518 transmitted onward to the rest of the recording system before continuing to step 512. Otherwise, no message is transmitted onward to the rest of the recording system and processing continues directly to step 512. If the translation module determines at step 512 that the CTI message indicates that the call has been concluded, at step 520 the module removes the call record from the associated devices. The translation module then adds the call record to the list of recently completed calls at step 528. Completed calls are discarded (step 530) after they get too old (i.e., after a predetermined number of recorded calls, or a given time period after the original recording of the call). Processing then continues again from step 502 by receiving the next incoming message. If at step 512 the call has not been concluded, the completed calls are discarded (step 530) after they get too old. Processing then continues again from step 502 by receiving the next incoming message.

If at step 504 the message is an SMDR message, the translation module at step 508 scans the list of recently completed calls. At step 510 the translation module calculates confidence factors for the recently completed calls by using the formula:

$$\text{Confidence Factor} = \sum_i ((\text{Match Quality})_i * (\text{Weighting Factor})_i)$$

If any matches are found (step 516), and more than one match is found (step 522), the match with the highest confidence factor is used (step 526). If only one match is found, that match is used (step 524). At step 540, the trunk channel information is extracted, and at step 544 the call record is updated within the list of recently completed calls. The call record is transmitted at step 548 to the rest of the recording system. At step 550 the call record is discarded from the list of recently completed calls. Completed call are discarded (step 530) after they get too old. If no matches were found at step 516, the completed calls are discarded (step 530) after they get too old. Processing then continues again from step 502 by receiving the next incoming message.

As shown in FIG. 6, the CTI Server can be viewed as a set of logically distinct layers that deal with translating and distributing CTI events. Starting from the bottom of the picture, CTI events flow from a PBX in its proprietary format to Dialogic CT-Connect middleware 640 another API layer 650 or custom interface layer 660 that each provide partial normalization of the data. This helps to reduce the complexity of the "translation" job, since there are fewer

US 6,249,570 B1

15

APIs than individual PBX types. But since one object of the subject system is to retain the flexibility to integrate with a variety of third-party CTI vendors (e.g., Dialogic, Genesys, etc.) there is another layer 670 above the API or custom interface layer to complete the job of "translation." The final result after passing through this "normalization" layer is that all of the CTI events are in a single, common, integrated data format.

Once the CTI events have been converted to a normalized format, the CTI Server can address its other mission of distributing (routing) the messages. The distribution layer 680 examines each message to determine what other recording system components need to receive it, and then sends a copy of the event to the appropriate destination(s).

This logical separation of responsibilities used in a preferred embodiment simplifies the programming required to implement the subject system. Translation modules do not need to know anything about other recording system components, and they can focus on dealing with a single specific PBX or vendor API layer. Likewise, the distribution module will not need to know anything about specific PBX or vendor API layers, and it can focus on making routing decisions and communicating with the rest of the recording system.

FIG. 7 illustrates how, in addition to telephony events, the CTI Server 710 used in accordance with the present invention is responsible for supplying certain metadata regarding agent events to the System Controller, which is part of the Scheduling & Control Services 130 shown in FIG. 1. This information, which generally includes agent ID, extension number, logon and logoff time, etc., is obtained when available from the various PBXs, ACDs, and turret systems. The agent events delivered to the System Controller 130 enable a map to be maintained of the extension number(s) where a real person can be found, at a given date and time. This information enables a browser application to intelligently associate some of the previously recorded calls even if a person was using different telephone sets according to a 'free seating' plan. The CTI Server 710 also keeps a local cache of the agent information, so that agent information can be included when sending the telephony events to the Call Record Generator 150.

The physical layout of the CTI Server used in a specific embodiment is shown in FIG. 8. With reference to FIG. 1, the translation modules are implemented by separate programs, such as CtiCtc.exe 406, which encapsulate the details on converting a specific PBX interface or vendor API layer into a normalized format. The distribution module is preferably implemented by a single program, CtiServ.exe 820, which includes the main processing and routing logic for the CTI Server.

As noted, the translation modules of the CTI Server convert proprietary-format CTI information into a normalized format. In accordance with a preferred embodiment, this is done in several layers within the program. The information is first converted by Dialogic's CT-Connect software into the CTC-API format, and then the conversion to the generic format used by the other components of the recording system is completed by the translation module CtiCtc.exe. Once the data is converted, it is transmitted to the distribution module (CtiServ.exe) by using a distributed communications method such as DCOM. Component Object Model (COM) is a Microsoft specification that defines the interaction between objects in the Windows environment. DCOM (Distributed Component Object Model) is the network version of COM that allows objects running on different computers attached to a network to

16

interact. An alternate embodiment of the CTI Server utilized Microsoft Message Queue (MSMQ) technology as the means to carry messages among the system components, instead of the original DCOM method used by CtiServ.exe, and those skilled in the art would appreciate that a variety of additional data communications technologies are also suitable to this role.

The translation module and the distribution module of the CTI Server can be located on different machines, if desired. There can be multiple translation modules running in the system—one for each PBX or CTI middleware environment. There can also be different types of translation modules, with one version for each interface or API layer. As depicted in FIG. 8, CtiCtc.exe deals with the Dialogic CT-Connect API, and there are 3 copies of this program running to handle the PBXs. If other types of APIs are used, there would be other programs for these various interfaces. All translation modules contribute data upward to the distribution module in a single, common, normalized format. An example of a version of CtiCtc.exe configured to work with a Lucent Telephony Services interface (and thus called CtiLts.exe instead of CtiCtc.exe) is shown in FIG. 9. The modules which are common to both versions of the program are shown in FIG. 9 as shaded gray. The unshaded modules represent those portions of the program that necessarily vary between CtiCtc.exe and CtiLts.exe, due to the differing input parameters and data structures used by both systems.

Again with reference to FIG. 8, the distribution module (CtiServ.exe) receives and collects all the CTI events from the various translation modules. Then it puts the events into a single inbound queue 830 for processing by a main control thread 835. After the events are processed, they are separated into individual outbound queues 840. Finally, the events are sent by various delivery threads 850 to the CRG components within different Voice Servers. The main processing thread 855 (WinMain) is deliberately isolated (decoupled) from the inputs and outputs to ensure that delays in transmitting or receiving data will not impact the overall performance of the CTI Server.

FIG. 11 shows how the CTI server in accordance with a specific embodiment consists of several threads that implement three distinct layers of processing (data collection 1110, data normalization 1120, and message emission 1130). FIG. 12 illustrates the processing steps of these layers. The dashed lines indicate message flow between threads, whereas the solid lines indicate program logic flow. The CTI translation modules are thus internally separated into 3 major sub-tasks: (1) data collection from the input source (PBX, CTI middleware, etc.); (2) normalization of the data to a common format; and (3) communications with the system platform.

In a data collection layer, the initial step 1210 is to open the connection to the CTI data source. At step 1214 the layer receives a CTI event, and at step 1216 posts the CTI event to the Message Queue 462 (see FIG. 4). If at step 1218 a shutdown is in progress, the connection to the CTI data source is closed at step 1220, and at step 1222 data collection is ended. If at step 1218 a shutdown is not in progress, the CTI connection remains open (step 1212).

At step 1228, the data normalization layer receives a CTI event from the Message Queue 462. The data normalization layer updates the telephony model at step 1230. See FIG. 13 for a more detailed explanation of the updating of the telephony model. At step 1231, the call state is posted to the Message Queue, if necessary. At step 1232 completed calls are discarded from memory after they age beyond a configurable time limit. At step 1233 the "hang-up" routine is

US 6,249,570 B1

17

called to update the telephony model for held or bumped calls after they age beyond a configurable time limit. At step 1234, if a shutdown is in progress, the data normalization layer checks the inbound message queue at step 1236. If the message queue is empty, data normalization is ended (step 1238). If the message queue is not empty at step 1236 or if there is not a shutdown in progress at step 1234, the data normalization layer goes to step 1226 and waits for the next CTI event to arrive.

The message emission process begins with opening a connection to a target platform, such as the da Vinci or SymphonyCTI recording systems at step 1240. At step 1244, the message emission layer receives the call state from the message queue 462. At step 1246, the call state data is converted into a platform-specific format. At step 1248, the message emitter sends the message to the target platform. At step 1250, if a shutdown is in progress, a check is made at step 1252 for whether the inbound message queue is empty. If the inbound message queue is empty, message emission is ended at step 1254. If the inbound message queue is not empty at step 1252, or if there is not a shutdown in progress at step 1250, the message emission layer, at step 1242, maintains the open connection to the target platform and awaits the next call state transmission.

Master Call Record

The CTI Server sends "Call Event Records" onward to the recording platform. These messages provide details on the start and end of calls, as well as significant transitions that affect the lists of participants for the calls. The list of participants is cumulative, and information regarding participants is retained for the entire duration of the call even when some participants in the list may have dropped off from the call. If a participant rejoins the call, a new, separate entry will be created to reflect that change within the participant list. The following table shows the fields contained within these messages.

CTICallEvent		
Name	Type (max length)	Description
Version	WORD	Version number of this message format, for reverse compatibility.
MessageID	GUID	Unique ID for this message instance
RecorderNode	WORD	A number that identifies a particular Voice Server
RecorderChannel	WORD	A number that identifies a recording input channel on a Voice Server
EventType	BYTE	Indicates if this event added (0 × 01) and/or dropped (0 × 02) participants in the call.
EventReason	BYTE	Indicates if this call was affected by a normal (1), conference (2), or transfer (3) telephony event
CTICallRecId	GUID	Unique ID pertaining to entire call (CTI server provides the same ID for a call that is transferred, conferenced, etc)
CallDirection	BYTE	Indicates call origin - outbound (0 × 12), inbound (0 × 21), internal (0 × 11), or unknown (0 × 44)
RingLength	WORD	Seconds between the first ring signal and going off-hook (picking up the phone)
DTMFCode	String*(50)	DTMF codes entered during the call

18

-continued

CTICallEvent		
Name	Type (max length)	Description
ApplicationData	String*(32)	Character array dedicated to information the switch may provide along with the call (e.g., account number)
CallingParty	WORD	Index number of the calling party within the participant list. Normally this is zero.
CalledParty	WORD	Index number of the called party within the participant list. Normally this is one.
PBXCallRecId	DWORD	Number provided by the PBX to identify this call.
NumberOfParticipants	WORD	Count of participants in the following array.
ParticipantList	Vector*	Array of PartyListElement describing all participants involved in the call

*ObjectSpace data types

ObjectSpace is a set of C++ class libraries provided by ObjectSpace, Inc., that define useful general-purpose data structures including representations of strings, time values, and collections of objects (such as vector arrays or linked lists). These class libraries are implemented in a way that supports a wide variety of computer operating systems. Those skilled in the art will appreciate that many alternate implementations for such data structures are suitable for this role.

CTIPartyListElement		
Name	Type	Description
AgentID	String*(24)	Registered ID of person, typically used for "free seating" call center environments
Number	String*(24)	Telephone number of this participant (e.g., ANI, DNIS, Dialed Digits)
Console	String*(10)	Seating position that can consist of one or more stations.
Station	String*(10)	Unique telephone set, possibly with multiple extensions.
Extension	String*(6)	Internal line number of the participant
SwitchId	WORD	Number of the switch (PBX, ACD, or turret system) which is handling the conversation
TrunkID	WORD	Identification of trunk line which is handling the conversation
VirtChannel	WORD	Identification of trunk's channel (time slot) which is handling the conversation
LocationReference	BYTE	Describes the location of participant with respect to the switch - can be internal (1), external (2), or unknown (3)
StartTime	time_and_date*	Time participant joined the call
EndTime	time_and_date*	Time participant left the call
ConnectReason	BYTE	How participant joined the call: norm start of call (1), being added to a conference (2), or receiving a transferred call (3)

US 6,249,570 B1

19

-continued

<u>CtiPartyListElement</u>		
Name	Type	Description
DisconnectReason	BYTE	How participant left the call: normal end of the call by hanging up (1), dropping out of a conference (2), transferring away a call (3), or call ends by another party hanging up (4).
Changed	BOOL	Indicates if recent change in CTI message.

*ObjectSpace data types

For external participants, only the fields Number, SwitchName, TrunkID, VirtChannel, LocationReference, StartTime, EndTime, ConnectReason, and DisconnectReason will be applicable. For internal participants, all fields may be applicable. Unused string fields will be null terminated. Unused number fields are set to zero. Each call event record will contain at least two participants in the list. These two participants are the original calling party (0) and called party (1) and will appear within the list in that order respectively.

Note: The data field "Number" will be filled in a variety of ways, depending upon the type of participant and direction of the call.

Participant Type	Call Direction	Number Field
External participant	Inbound Call	ANI
External participant	Outbound Call	Dialed Digits
Internal participant	Inbound Call	DNIS or Extension
Internal participant	Outbound Call	Extension
Internal participant	Internal Call	Dialed Digits or Extension

The CTI Server sends "Agent Event Records" onward to the recording platform's System Controller to convey information when an agent logs on/off at a particular location. The following table shows the fields contained within these messages.

<u>CtiAgentEvent</u>		
Name	Type (max length)	Description
Version	WORD	Version number of this message format, for reverse compatibility.
MessageID	GUID	Unique ID for this message instance
EventType	BYTE	Indicates if this event pertains to either a logon (1) or logoff (2).
LocationType	BYTE	Indicates if this event pertains to a location type such as a console (1), station (2) or extension (3).
AgentID	String*(24)	Registered ID of person, typically used for "free seating" call center environments
SwitchId	WORD	Number of the switch (PBX, ACD, or turret system) where the agent connected.
Console	String*(10)	Seating position that can consist of one or more stations.
Station	String*(10)	Unique telephone set, possibly with multiple extensions.
Extension	String*(6)	Internal line number of the participant

20

-continued

<u>CtiAgentEvent</u>		
Name	Type (max length)	Description
StartTime	time_and_date*	Time that the agent logged in.
EndTime	time_and_date*	Time that the agent logged out.

*ObjectSpace data types

Within any given "Agent Event Record", only one of the following three fields will be applicable: Console, Station, or Extension. The actual mapping is determined by the LocationType. Unused string fields will be null terminated. Unused number fields are set to zero.

It will be appreciated that the general principles behind the method described above are suitable not only for associating and combining real-time CTI data with the trunk channel information from an SMDR message, but also for any situation where a mixture of information is being provided from two or more sources and there is a need to gather and merge the information to get a more complete picture of what is actually happening in the system. The disclosed method could easily be adapted by those of ordinary skill in the art to situations in which the mapping or association between the multiple sources of information is "weak" and prone to ambiguity. While this method does not make the potential ambiguity disappear, it helps to define a quantitative set of rules for making a judgement call on when a match is "good enough" to act upon. While human beings are often capable of making such judgement calls intuitively, computers need a specific set of instructions in order to act in a repeatable and reliable fashion upon the input data.

Previous recording systems that made use of CTI to collect enhanced search information mimicked the event-oriented interfaces provided on the data links from a PBX. Individual database records were constructed on a 1-to-1 basis for the events occurring during the total lifetime of a phone call. The interpretation of the series of events was left to the end user. Associations between related events were made difficult in certain cases because the call identification numbers given by a PBX may change after a call has been transferred or conferenced, or the numbers may be recycled and reused over time. Following and tracing the history of events for a complete call from the perspective of the external customer could require much manual and repetitive searching. Playing back the entire set of audio recordings from the start of that customer's interaction with the business, to the ultimate conclusion of that customer's transaction, could also require additional repetitive manual requests to play back the individual recorded segments within a call that was transferred or conferenced.

To resolve this problem, the CTI server of the preferred embodiment maintains and accumulates information within a data model of telephony activity. FIG. 10 depicts the key elements of the data model. This consolidated information is shared with the rest of the recording system when parties join or leave a call, thereby eliminating the need for downstream components to store or interpret the individual CTI events occurring during a call's lifetime.

During the active lifetime of a call, real-time information is accumulated within a historical call record that tracks each participant within the call. At certain key points during the accumulation of data, whenever a party joins or leaves the conversation, the call record is transmitted onward to allow the rest of the recording system to process the information accumulated to that point. Upon the conclusion of the call,

US 6,249,570 B1

21

the CTI server of the preferred embodiment retains a copy of the call record for a configurable time interval before discarding it from memory. This delay allows for the arrival of the SMDR data.

The call records are organized into a two-tiered hierarchy of calls and participants. Certain data fields that apply globally to the entire call are stored at the upper level. Most data fields, however, apply only to a specific party involved within a call, and are stored at the lower level. Individual participants can have identifying information (such as extension number, agent ID, telephone number via DNIS/ANI/CLID, trunk and channel) along with time-stamps and reason codes for the entry and exit from participation in the telephone call. Reason codes include initial start, transfer, hold, resume, conference add/drop, and hang-up.

The currently active call on each telephone set being monitored is maintained within a storage area 1020 of the data model. Also, the data model provides for an open-ended list 1040 of calls that may be "on hold" (and therefore not associated with any telephone set). There is also a list 1030 that can be used temporarily for calls when they are in a state of transition during transfers, queuing or re-routing, for the brief period of time when an active call is disassociated from its original telephone set but not yet associated with a new telephone set. Finally, there is a list 1050 of recently completed calls that is used to await additional information that might be provided from a SMDR message.

This complete set of data structures is replicated independently for each CTI server that monitors a separate PBX within the overall call center environment.

The call-centric structure and the list of participants facilitate a common framework for modeling the various types of complex call scenarios that may occur during the life of a call, far beyond the simplest example of a basic two-party telephone call. Moreover, the recording units can link references (i.e., logical pointers) to the audio recordings for a portion of the call, so that these audio sections are associated with the total history of the logical telephone call. Each call record can be linked within the database to an open-ended list of references, which provides: the name of a Voice Server; the name of a .WAV file containing the audio recording; the offset within the .WAV file to the start of the recording segment; the start time of the recording segment; and the duration of the recording segment.

Rather than relying exclusively upon the call identification number assigned by the PBX, the CTI server of the preferred embodiment obtains a Globally Unique Identifier (GUID), that is generated at the software's request by the underlying Microsoft Windows NT operating system, and uses that GUID to identify the call uniquely within the recording system's memory, online storage database, and offline storage archives. The GUID is initially requested at the start of the call. While the call remains active, the CTI server maintains a record of both the call identification number assigned by the PBX, and the GUID assigned to the call by the software of the preferred embodiment. When a CTI event arrives, the system searches the telephony model to find a matching call record for the PBX-assigned call identification number. At transition points during a call's lifetime, such as when it is transferred or conferenced, the PBX typically provides the old and new identification numbers together in that single transition event. In these cases, after locating the matching call record, the software of the preferred embodiment updates its record of the call identification number now being used by the PBX while retaining the originally allocated GUID value. In this way, the same GUID identifies the call throughout its lifetime, even while

22

the PBX call identifier may be changing. The long-term uniqueness of the GUID value is also useful if the PBX recycles and reuses previously assigned call identifiers. It further helps in dealing with calls within a multiple PBX environment. While another PBX may coincidentally use the same call identification number, a different GUID is assigned at the start of each individual call, thereby avoiding a conflict within the telephony model.

As shown in FIG. 11, the CTI server consists of three distinct layers. Each layer actually runs in a separate thread of execution, and communicates with the other layers through shared memory, control semaphores, and message queues. The first layer 1110 is responsible for gathering input from the PBX data link(s), and there can actually be several threads running to provide better throughput capacity or to handle multiple diverse input sources (e.g., SMDR and real-time CTI messages). After saving the clock time when a message is received, the first layer 1110 places the message into a queue for subsequent processing by the second "analyzer" layer. The second layer 1120 is responsible for updating and maintaining the telephony model within the memory of the CTI server, and for deciding when to send copies of call records onward to the rest of the recording system. When a call record needs to be sent onward, the call record is placed into a message queue for subsequent processing by a third "message emitter" layer 1130, which is responsible for communications with other components of the overall recording system. This separation of layers gives the CTI server the flexibility to process its input and output sources in a de-coupled fashion, so that any delay in one area of communications does not affect the processing of another area. In a sense, the design approach provides a virtual "shock absorber" so that bursts of input traffic, or temporary lag times in communicating with other parts of the recording system, can be tolerated without loss of data or incorrect operation of the system.

The call records saved within the telephony model also include a record of the last state of the device as reported by the PBX. This information is used by the analyzer to run state machine rules, in order to select a handler routine for a subsequent message. The CTI server uses the previous state of the device (e.g., ringing, answered, and so forth) along with the current state of the device to select a handler routine from a matrix of potential choices.

The analyzer layer is of particular interest, since it is responsible for updating and maintaining the data model of telephony activity. Its overall program logic flow is illustrated in FIG. 12 and the subroutine called at step 1230 is shown in further detail by FIG. 13. This program logic is described below.

1. Receive a CTI event from the message queue at step 1228.
2. Enter the subroutine at step 1230 to update the telephony model. Referring now to FIG. 13, search the data of model of telephony activity, to find a matching record at step 1322 with the same monitored device (i.e., telephone set).
3. If the PBX-assigned call identification number does not agree, search for a matching record in the lists of calls on hold, in transition states, or recently completed. If a match is then found, move the call record on the affected device to the list of calls in transition states, and move the matching record to the monitored device.
4. At step 1324, use the previous state as recording within the telephony model, along with the new state reported in the CTI event, to select the appropriate handler routine at step 1332 from a matrix of choices. The handler routine will be one such as those described below.
5. At step 1340, run the steps of the handler routine. This will commonly include steps to save at step 1342 information

US 6,249,570 B1

23

from the CTI event into the call record, to update the call-related portion of the Object Status, if necessary (step 1344), to update Participants within the Object Status, if necessary (step 1352), to run additional action methods or handler routines for other affected telephony objects, if necessary (step 1348), and to post Object Status to the message Queue for the Emitter to a target platform (step 1354).

6. At step 1360, returning to FIG. 12, at step 1232, discard completed calls within the data model of telephony activity, if they have aged beyond a certain re-configurable time limit.
7. Call the "hang-up" routine at step 1233 for any held call that have aged beyond a separate re-configurable time limit. Likewise, call the "hang-up" routine for any calls marked in transition, which have aged beyond another separate re-configurable time limit.
8. Continue again from the beginning of this logical program flow at step 1226.

The following description lists processing steps for various handler routines that may be called in response to certain event types using a decision matrix based upon past and current state information.

Handler Routines

Ignore: adjust state based on CTI event

DialTone: save the initial start-time of the call
save the original dialed number, if available
adjust state based on CTI event

RingIn: adjust state based on CTI event
event time-stamp when ring occurred
clear call record

set inbound, outbound, internal
Answer: adjust state based on CTI event
compute total ringing duration
fill in call record with calling party & called party
generate START message to recording system

Abort: adjust state based on CTI event

clear timers & original dialed number
Hang-Up: adjust state based on CTI event
update call record to stop all parties
indicate which party actually hung up on the call
generate STOP message to recording system

RingOut: adjust state based on CTI event
time-stamp when ring occurred (i.e., now)
clear call record

set inbound, outbound, internal
compute total ringing duration (i.e., zero)
fill in call record with calling party & called party
generate START message to recording system

Hold: adjust state based on CTI event
stop participant placing the call on hold
add new placeholder participant for HOLD
generate TRANSFER message to recording system
move call record to hold area

fill device slot with a new empty call record
Resume: if device slot not idle, move call record to transition list

move matching call record from hold area to device slot
adjust state to "active"

stop the placeholder participant for HOLD

add new participant for telephone set that resumes the call
generate TRANSFER message to recording system

24

Conference: if call record found in hold area,
if device slot not idle, move call record to transition list
move matching call record from hold area to device slot
adjust state to "active"

stop the placeholder participant for HOLD
add new participant for telephone set that resumes the call
generate TRANSFER message to recording system
adjust state based on CTI event

add new participant for telephone set that is added via conference
generate CONFERENCE-ADD message to recording system

Transfer: if call record found in hold area,
if device slot not idle, move call record to transition list
move matching call record from hold area to device slot
adjust state based on CTI event

stop the participant leaving the scope of the call (either a device or HOLD)

add new participant receiving the transferred call
generate TRANSFER message to recording system

ConfDrop: adjust state based on CTI event

stop the participant leaving the scope of the call
generate CONFERENCE-DROP message to recording system

OpAnswer: adjust state based on CTI event

re-compute total ringing duration
correct the affected participant entry in the call record
generate CORRECTED message

DestChanged: clear call record
the call will be processed via a subsequent CTI event

The following step-by-step description describes the same call scenario as in FIG. 3, but with emphasis on the data model of telephony activity.

1. A real-time CTI message occurs describing that phone B is ringing, but not yet answered.

2. The "RingIn" routine is invoked.

3. The telephony model is updated with the time when ringing started (for use later in measuring ring duration) and the call direction. These facts are stored with device B 340.

4. A real-time CTI message occurs describing the start of the call between A 335 and B 340.

5. The "Answer" routine is invoked.

6. The telephony model is updated to reflect the initial 2 participants (A and B) started normally at t0 310.

7. A copy of the call record is sent onward to the rest of the recording system.

8. The call record is retained within the telephony model, associated with device B 340.

9. A real time CTI message occurs describing that B 340 placed the call on hold.

10. The "Hold" routine is invoked.

11. The telephony model is updated to reflect that B 340 transferred the call to HOLD 345 at t1 315. (This information is accumulated with the information previously gathered at t0).

12. A copy of the call record is sent onward to the rest of the recording system.

13. The call record is removed from device B 340 within the telephony model, but kept in a list of held calls.

14. A real-time CTI message occurs describing that B 350 returned to the call and invited C 355 by conferencing.

15. The "Conference" routine is invoked.

US 6,249,570 B1

25

16. The call record is moved within the telephony model from the list of held calls back to device B 350.
17. The telephony model is updated to reflect that HOLD 345 transferred the call back to B 350 at t2 320. (Note that information is accumulated with the information previously gathered at t0 and t1).
18. A copy of the call record is sent onward to the rest of the recording system.
19. The telephony model is updated to reflect that C 355 joined the call as a conference participant at t2 320. (This information continues to be accumulated with previously gathered information).
20. A copy of the call record is sent onward to the rest of the recording system.
21. The call record is retained with both devices B 350 and C 355 within the telephony model.
22. A real-time CTI message occurs describing that C 355 dropped out of the call.
23. The "ConfDrop" routine 386 is invoked.
24. The telephony model is updated to reflect that C dropped out of the conference at t3. (This information continues to be accumulated with previously gathered information).
25. A copy of the call record is sent onward to the rest of the recording system.
26. The call record is removed from device C within the telephony model, but retained with device B.
27. A real-time CTI message occurs describing that A terminated the call.
28. The "Hang-Up" routine is invoked.
29. The telephony model is updated to reflect that A stopped normally and B stopped because the other party hung up at t4 330. (This information continues to be accumulated with previously gathered information).
30. A copy of the call record is sent onward to the rest of the recording system.
31. The call record is removed from device B 350, but kept in a list of completed calls.
32. A SMDR message occurs summarizing the call in its entirety.
33. The list of completed calls is searched to find a match, and the appropriate call record is retrieved.
34. The call record is updated with the trunk channel information from the SMDR message.
35. A copy of the call record is sent onward to the rest of the recording system.
36. The call record is removed from the list of completed calls.

FIG. 14 depicts the flow of information within the remainder of the recording system. The same enhanced search information S1 1412 is provided by the CTI server to all of the recording units involved in handling a portion of the call. Even if a call is transferred to another telephone set, which is attached to an input channel on a different recorder, the entire call will still remain associated as one entity within the system. Each recorder maintains a local copy of the audio sections V1 1416, V2 1420, and V3 1424 that it obtained during the call, along with a complete call record containing search information S1 1412 which contains the two-tiered call and participant model. The search information is copied to a central database server 1450, along with references (i.e., logical pointers) to the original audio recordings VR1 1428, VR2 1432, and VR3 1436. When a user searches for a call, the search results 1465 will include the complete call record S1 1412. By using the audio references the playback software can reassemble the complete audio for the original call, including sections possibly obtained from different physical recording units.

26

The general principles behind the method described above would be suitable, not only for representing the complete history of telephone call's lifetime, but other forms of multi-party communications. This may include certain forms of radio traffic that have an associated data link, which provides "talk group" identification numbers (or similar types of descriptive search data in relation to the audio traffic).

Call Record Generator

The Call Record Generator (CRG) in accordance with the present invention performs the function of combining voice and data into call records. It performs this function at or near real time. The CRG, when combined with the metadata normalization module CTI Server, makes up a system that can be used in current and future communication recording products.

The CRG is responsible for collecting data from different sources with respect to portions of a call on various recording input channels, and merging them together into a unified call record. One of these sources is the recorder that creates the files containing media. Another source provides metadata describing the when, who, why and where information of a call. This call record metadata comprises the start and stop times of a segment within a call, as well as CTI data such as telephone numbers and agent IDs. These metadata sources include but are not limited to Telephony switches and Trunked Radio servers. The CRG depends upon the CTI Server to normalize data from these sources.

FIG. 1 illustrates the relationship between the CRG and the rest of the system. Since call records are an essential part of the recording system, there is one CRG dedicated to each recorder and physically located in the same Voice Server. If other system components become inoperable, call record generation will remain functional (albeit at a reduced level).

The CTI server supplies switch events to the appropriate recorder indicating either the status of calls or providing data for population. The CTI server provides, along with call record data, the association between the recorder location (i.e., Voice Server and recording input channel number) and the switch connection point. The switch connection point is described as either the extension for extension side recording or the Trunk ID/virtual channel (TDM time slot) for trunk side recording. In addition to this mapping, an agent identification will be supplied for agents currently associated with this call. The recorder location, switch identification and corresponding agent are stored in the call record. The CRG is designed to work with many different configurations of the disclosed system. These configurations include: systems without CTI Servers; systems with Real-time CTI Servers; systems with non-Real-time CTI Servers; recorders with analog inputs; recorders with digital inputs; recording on the trunk side of the telephony switch; and any combination of CTI Servers, Recorder inputs, and recorder positions mentioned above.

Due to the non-standard operation of telephony switches and flexibility requirements of the recording device, the CRG must handle event data arriving in different chronological order. In accordance with a preferred embodiment, it accomplishes this by requiring all events to indicate time of occurrence and maintaining a history of them. A call record can be created solely from either event sources but when both are present, call records are generated using recorder information together with CTI data.

It is clear that the use of different data sources and non-synchronous messages, as required to support various alternative configurations of the overall system, add considerable complexity to the CRG. For example, with the many

US 6,249,570 B1

27

different objects supplying information for a particular call, the messages from each can be received in any order. The CRG must be able to accommodate this requirement. In some configurations, objects supply redundant information to the CRG. The CRG provides a mechanism for selecting which information will populate the call record.

In the most basic mode of operation, the CRG has no CTI input and is recording solely on VOX events from the recorder controller (the term "recorder controller" is used interchangeably herein with "Audio Recorder"; both terms refer to the software that primarily directs the processing of the audio data). VOX is Dialogic Corporation's digital encoding format for audio samples. This term is also sometimes used to refer voice-activated initiation of recording, a process that conserves storage space since a continuous recording process would include periods of silence. These VOX events mark the beginning of energy activity on a phone line and are terminated by the lack of activity. With this approach, an actual phone call may include several call records. To address this problem, the recorder waits a configurable holdover period while silence is present before terminating an active VOX clip (the term "Recorder" is used interchangeably herein with the term "Voice Server"; both terms refer to the physical recording server). The goal is to concatenate parts of a phone call where gaps of silence exist. The solution lies in determining an appropriate holdover time so as to avoid merging audio from the next phone call if it occurs close to the end of the last call.

The next level of operation is where the recorder hardware can detect telephony signaling such as off hook and on hook. The CRG has no CTI input from the switch and is recording solely on events from the recorder controller, but these events mark the beginning and end of a phone call (off hook and on hook). The resultant call record reflects a phone call in entirety but lacks much descriptive data that accompanies switch data.

The highest level of operation involves the use of a CTI Server. In this configuration, the CRG receives recorder events as well as CTI events. Since CTI events give the CRG a description of the entire phone call, information obtained from them drive the creation of call records. Recorder data describing audio events are absorbed into the CTI call record whenever audio and CTI times overlap. With CTI events driving call record generation, non-audio based call records can be created.

Mixing of recorder and CTI data occurs by comparing ranges of time indicated. For example, a person whose telephone extension is being recorded is involved in a phone call for a given period of time. The recorder events indicating that audio was recording on the same extension during the same time period are associated with the CTI metadata for that phone call. Since the data from the CTI Server may arrive before or after the corresponding recorder events, the CRG maintains an independent history for each type of data.

For the case where CTI events arrive before the recorder events, the CTI events are added to the CTI history list. When the corresponding recorder events arrive, the CTI history list is swept for matching time ranges and associations are made when they occur. For the case where recorder events arrive before the CTI events, the recorder events are added to the recorder history list. When the corresponding CTI events arrive, the recorder history list is swept for matching time ranges and associations are made when they occur.

Previous recording systems stored voice data and metadata in separate locations. A significant disadvantage to this approach is that it is left up to the other software subsystems

28

to combine the information when required. This approach makes the work of other system features, such as playback and archiving to offline storage, more complicated and prone to error. By performing this "early binding" of the audio and CTI data in accordance with the present invention, such problems are avoided and the above desirable features are therefore much simpler to implement in a correct, robust fashion.

When attempting to playback media for a given call record, the playback mechanism must figure out where the audio for the call record exists and when determined, retrieve and locate the start time inside this media. The CRG places this media metadata in related tables, thus informing the playback mechanism what files are associated, their location, and what time ranges inside the file are available for playback.

Most communication systems require an archive mechanism to store large amounts of data that cannot be kept online due to capacity limitations. The CRG used in accordance with this invention assists with archiving by allowing both call record metadata and the media files to be stored on the same offline media. Current versions of recording systems store call record metadata and media files on separate offline media making restore operations more complicated.

For enhanced security purposes in a preferred embodiment, the CRG accesses media files associated with a call record through the use of media segmentation. A media segment includes, in addition to a media filename and location, a start time and duration inside the media file. Media segmentation is necessary when creating CTI based call records since a call record may involve many recording locations throughout the life of the call. The specified time range isolates a portion of the media file that can be accessed through this call record. This feature is very important when there are many call records located in one media file. A user attempting to play back media of a call record, to which he has the permission for access, may or may not have permission to play back other call records sharing the same physical file.

The Call Record Generator is responsible for merging CTI search data and a multitude of voice recording segments together into a single manageable unit of data. This software includes a flexible receiver algorithm to allow voice and search data to arrive in either order, without requiring one to precede the other. Once combined, the call record can be managed as a single entity, which greatly simplifies and reduces the work necessary to perform search, retrieval, and archival operations. This approach also offers a more natural and flexible framework for controlling security access to the recordings, on an individual call basis (or even on selected portions within a call).

As shown in FIG. 15, a recording unit operating with only voice signaling to guide the creation of its call records could make a number of fragmented audio segments. When the recording unit is supplied with CTI search data giving a complete history of the call's lifetime, and when it is designed to merge the CTI search data and audio segments into a combined unit of Voicedata™, the results can simplify and reduce the work necessary for a user to obtain a desired call from the system. Several audio segments can be grouped together, and can be understood by the system as being part of the same logical telephone call. It is also possible that a single audio segment was recorded, even though parts belong to separate telephone calls, because the delay between stopping the first call and starting the second call was very brief. Without a sufficient silence gap, it may appear to the voice recording unit that this was a continuous

US 6,249,570 B1

29

segment of audio, rather than belonging to two separate calls. When the CTI search data is merged with the audio segments, the system can use this information to recognize when an audio segment should be split and divided between two logically distinct calls.

The purpose of the Call Record Generator (CRG) is to collect information describing multimedia data and store it in a central location. The CRG produces Master Call Records (MCRs) that encapsulate information describing a phone call as well as the location multimedia that is associated with it. This description data comes from a multitude of sources including but not limited to a Voice Server and CTI Server. Likewise, the design of the system envisions that there will be a number of possible input sources for audio recording.

Whatever the means for collecting CTI information, it is communicated to the rest of the system in a common, normalized format. The CTI information is passed from the translation modules to a message router. From that point, copies of the information are sent to the scheduling and control services and to the CRG for the appropriate recorder (s). The scheduling and control services are responsible for starting and stopping the audio recorder, according to pre-defined rules that are dependent upon time and CTI information. The CRG is responsible for merging the audio recording with the CTI information to determine the temporal boundaries of the call and prepare the Voicedata for storage.

The user workstation typically searches and retrieves records from the Voicedata storage, and then obtains audio for playback directly from each recorder's private storage area. The user workstation can also be used to monitor "live" conversations by communicating directly with the recorder. The user workstation can also control the audio recorder indirectly by manipulating the rules used by the scheduling and control services.

In the preferred embodiment, the user workstation has software that is configured to display a graphical user interface (GUI) such as that shown in FIG. 16. The GUI in FIG. 16 uses the information compiled in the Master Call Record to generate a graphical representation 1610 of the call, as well as displaying the call record information in alphanumeric form in a table 1620. Further, when the call is played back, the displayed segments in the graphical representation are highlighted to indicate the portion of the call being played back. For example, in FIG. 16, if the entire call is played back, when the portion of the call that occurred between 6:20:08 AM and 6:55:31 AM is played back the bars 1632, 1634, and 1636 are highlighted from left to right as the call is played back. Thus, as the part of the call that occurred at 6:55:31 AM is reached, bar 1634 is fully highlighted, and bars 1632 and 1636 are highlighted starting from the left and extending to those points on bars 1632 and 1636 that are directly above and below the right-hand endpoint of bar 1634. After the played back call reaches the part that occurred at 6:55:31 AM, the bar 1638 begins to be highlighted starting at the left endpoint. When the part of the call that occurred at 7:10:22 AM is reached, the bar 1636 is fully highlighted. At that point, the bars 1632 and 1636 are highlighted from their left-hand endpoints and extending to points directly above the right-hand endpoint of bar 1638. The process continues as long as the call is being played back, until bars 1632, 1634, 1636, 1638, 1642, and 1644 are completely highlighted.

In alternate embodiments of the subject invention, playback of a portion of a call can be activated directly from the graphical view by mouse-clicking or by selecting from a

30

pop-up menu; circular "pie-charts" show the percentage of time for each party involved during the lifetime of the call; an animated vertical line scrolls along to indicate the progression of time when the call whose graph is being displayed is played back; and miniature pictorial icons are shown within the graphs to indicate start/stop reasons, type of participant, etc. All of these embodiments are enabled by the data contained in the Master Call Record.

As a method of managing complexity, the preferred embodiment of the system uses data abstraction to isolate the internal details of certain structures to those components which need to operate directly upon them. Information is organized by the collectors (or producers) of that data, into a digested form that is more easily usable by the applications which need to retrieve and process the data.

For example, the CTI translation modules supply normalized records to the rest of the system in a common shared format, rather than exposing the details of various different CTI links. The system data model is call-centric, containing a detailed cumulative ("cradle to grave") history, rather than event-centric, which would place the burden of work on the receiving applications. Likewise, agent information is session-oriented rather than event-oriented.

Whether collecting information from a CTI link, or recording audio from a telephone call, a fundamental design advantage for the system of the preferred embodiment that it operates virtually invisibly, from the end-user's perspective. The system architecture is designed to avoid any interference with the normal operation of a call center environment.

For example, the CTI translation modules are focused exclusively on collecting and normalizing information that is to be supplied to the rest of the system. Liability recording systems, and quality monitoring systems that use "service observance" techniques, do not require any active call control on the CTI links. Only the technique known as "dynamic channel allocation" requires active call control through CTI links to establish a "conference" or "bridge" session between the audio recorder and the telephone call participants. When active control is required to implement such a feature, it can be implemented through a new logically separate task, without significantly affecting the rest of the system design. For customers that have existing CTI infrastructure and applications, the system will not interfere with their existing operations.

The CRG is responsible for collecting data from the CTI Server, creating CTI-based call records, and attempting to match those records with existing recorded audio data. If the CRG receives CTI information indicating that audio data for the same call resides on two or more recorders (for example, due to a transfer), records will be generated for each portion with a common call record ID. This ID can later be used to query for all of the pieces ("segments") comprising the complete call. Each segment will identify the recorder that contains that piece of the call.

During playback, a player module connects to a program located on a Voice Server called the Playback server ("PBServer"). The machine name of the particular Voice Server which holds an audio segment is stored by the CRG in the call record table within the Voicedata storage, and is passed into the player module after being extracted by a User Workstation's sub-component known as the call record browser. A call record playback request is then submitted, which causes the PBServer to query for a specific call record's audio files located on that physical machine, open them, and prepare to stream the audio upon buffer requests back to the client software (the player module) on the User

US 6,249,570 B1

31

Workstation. If successful, a series of requests is then issued from the client, each of which will obtain just enough audio to play to a waveOut device while maintaining a safety net of extra audio in case of network delays. Upon a request to "move" within the scope of a call record, the PBServer repositions its lead pointer to the desired location and then begins passing buffers from that point. This series of Request and Move commands continues until the user chooses to end the session by shutting down the client-side audio player.

As used herein, the term "Call Control" refers to the part of the metadata concerning the creation and termination of call records. The term "Media" refers to the actual data that is being recorded. This term is used interchangeably with audio since the primary design of the CRG is to support audio recording. However, the CRG could apply to any data being recorded including multimedia or screen image data. The term "Metadata" refers to informational data associated with multimedia data describing its contents. The term "Call Participant" refers to an entity that is involved in a phone call. There are at least two participants involved in a call; namely the calling and called parties. Participants can consist of people, VRUs, or placeholders for parties being placed on hold. The term "Recorder Participant" refers to a participant in the MCRs Participant list who is located at the same connection point on the Switch to which the recorder input channel is connected. In accordance with the present invention, there can be more than one Recorder Participant associated with a call record since participants can enter and leave many times in a call. For any given recorder channel, there can only be one matching Recorder Participant active (not disconnected) at any given time across all call records associated with that channel. A "VOX-based Master Call Record contains information contributed by events from the Recorder alone, in the absence of data from a CTI Server. A VRU is a Voice Response Unit: an automated system that prompts calling parties for information and forwards them to the appropriate handler.

Once a recorder channel becomes involved in a phone call, it will be associated with all subsequent CTI events pertaining to the same call. This occurs even if the recorder location is no longer involved in the call. As an example, consider a phone call involving a transfer. FIG. 16A shows the subject system containing a CTI Server 710 and Recorder 1640. A recorder channel 0 1650 is attached to the extension side to extension 0001 1622. A phone call is initiated from the outside by some agent "A" 1602 and initially connects to agent "B" 1608 at extension 0001 1622. Agent "B" 1608 places "A" 1602 on hold and transfers him to Agent "C" 1612 at extension 0002 1630. The CRG recording extension 0001 1622 would receive all update messages with regard to this call since he/she participated in the call. Descriptive information from the CTI Server 710 would look like that in table 1600 in FIG. 16B. Audio clips recorded while agent "B" 1608 was involved in the call are recorded in a VOX based call record as shown in FIG. 17. The three media files created from the conversation may overlap with the recorder participant (agent "B"). At some point, determined by the order by which recorder and CTI events are received, audio data information from the VOX call record is absorbed into the CTI MCR for the times the recorder participant is involved (see the results after the sweep of the VOX and CTI history lists). For this call record, audio recorded between times t_1 and t_4 is absorbed. Any remaining audio is left in the VOX MCR for possible absorption in other CTI MCRs adjacent in time to this one. Since extension 0001 in this call record is different from the

32

other participants in that it is associated with the same switch point as the recorder channel, he/she is referred to as the Recorder Participant. From time t_4 and on when the Recorder Participant is no longer involved in the call, CTI events are still received for that channel. This allows the system to supply information about the entire phone call involving extension 0001 that may be of interest to the customer.

Since the CRG must be prepared to handle messages from different components arriving in any order, it is designed to collect information in separate structures. Depending upon the operating mode of the CRG channel, call records are created from information collected in one or more of these repositories. The name given for these structures is Master Call Record (MCR).

The major components of the preferred embodiment contributing information for call records are the Recorder and the CTI Server. In alternate embodiments of the subject invention, other multimedia or screen image data may be provided to the CRG in order to be merged with descriptive metadata.

Recorder events are assembled into VOX MCRs identified by a unique sequence number. Individual events contain a sequence number identifying a specific structure to update (or create). For example, a recorder event would be used to indicate the beginning of a new audio segment. While that segment is active, other messages containing the same sequence number are used to add metadata to the audio segment. These update events include, without limitation: DTMF digit data; agent association information; change of audio filenames holding future audio data; selective record control; and ANI, ALI, DNIS information. DTMF is Dual Tone Multi-Frequency and refers to sounds emitted when a key is pressed on a telephone's touch-tone keypad; ALI is Automatic Location Identification, a signaling method that identifies the physical street address of the calling party and typically used to support Emergency 911 response. Finally, a disconnect message identifies the end of an audio segment.

Events received from the CTI Server are accumulated in CTI MCRs. Each event received from the CTI server contains a unique identifier. Events containing the same unique identifier are associated with the same CTI MCR. If any VOX MCR contains audio data that overlaps in time with Recorder Participants in a CTI MCR, then that audio data is transferred to the CTI MCR. If the absorption process causes all audio metadata for a VOX MCR to be consumed, the VOX MCR is deleted from the VOX list. Therefore, call records generated on the same channel will never have overlapping audio data. VOX MCRs containing leftover audio not absorbed by CTI MCRs are either be saved into the central database if of significant duration or discarded.

Data from a Master Call Record alone is processed into call record(s) that populate the system's central database. Thus, if the recorder channel is set up for VOX based recording only or if the CTI Server is down, VOX MCRs drive call record creation in the system. Otherwise, the CTI MCRs drive call record creation in the system.

The VOX and CTI MCR structures are maintained in two separate lists for each recording input channel. These are the VOX History List and CTI History List respectively. These lists represent a history of call activity sorted chronologically. The depth of the history list is driven by a configurable time parameter indicating the amount of history that should be maintained. By maintaining a history, the CRG tolerates events received in any order as long as received within the time boundaries of the history list. Some CTI Servers obtain data from SMDR type switches which report entire phone calls at the end of the call with a summary message.

US 6,249,570 B1

33

Maintaining a history buffer for VOX MCRs allows us to hold onto audio data for a period of time to allow later CTI summary messages to consume (absorb) the associated audio.

The MCR has status fields associated with them indicating its current state. At an installation involving real time CTI events, when a recording input channel receives a CTI event, it may indicate that a participant connected at the same telephony switch location as the recorder (Recorder Participant) is active in the call. The MCR is considered active as long as there is a Recorder Participant still active in the call. During this period, any new audio arriving on this channel is associated with the MCR. When a Recorder Participant leaves the call, the MCR becomes inactive. Since any Recorder Participant can become involved in the conversation at any given time through transfers or conferences, the MCR can transition into and out of active state many times throughout the phone call.

Another field in the MCR indicates the overall status of the call. This flag, called `m_bComplete`, indicates when the phone call is over. An MCR is considered incomplete as long as there is at least one participant still active in the call. When there are no participants active in a MCR it is considered to be complete. Therefore, calls created in real-time will start as incomplete and at some point transition into completed state. When an MCR enters complete state, a Closed Time variable is set to the current time. This time is used in maintenance of the History List. A closed MCR is allowed to stay in the History list for a configurable amount of time before it is deleted. During this window of time, events arriving out of timely order are allowed to update the MCR. Once this configurable amount of time expires, the MCR is updated in the local database, marked complete, and deleted from the History List.

When the CRG starts, it initializes, for each recording input channel, a location which identifies where it is attached to the telephony switch. Each recorder location contains status fields describing the state of the switch and CTI server involved. These fields are `m_SwitchStatus` and `m_MetadataServerStatus` respectively and are set to "down" state until an event is received that indicates otherwise. When a message is received indicating a change of state, all associated recorder locations are updated with the new state value. Any changes in operation are processed upon receipt of the next event for the channel.

Another configuration setting indicates what type of external sources are allowed to populate call records created on a record channel. This setting, `m_ExternalMetadataSource`, is set to zero when a record channel is to be driven by recorder events only. It is set to non-zero when external events are allowed to generate MCRs.

The CRG is able to react to a variety of situations that may arise. For example, when the CRG first initializes and a record channel is configured to receive CTI input, how are call records generated if the CTI server is not running? What if the CTI Server is running but the communication path to the recorder is down? The CRG must also be able to react to external parts of the system, that it normally relies on for input, being temporarily unavailable for periods of time. In accordance with a preferred embodiment, the CRG handles these situations by operating in different modes: Initial, Degraded, and Normal. These modes are applied individually to each channel in the recorder.

Initial Mode

When a recorder starts up, there can be a considerable amount of time before the rest of the system becomes operational. The CRG must be ready to handle events coming from the Recorder immediately after startup. Therefore, the CRG must be ready to accept recorder metadata without supportive information from the CTI

34

server. VOX MCRs are created from these recorder events and are stored in the VOX History List. When VOX MCRs are completed, they are made persistent in the Local Data Store.

The CRG system will remain in this mode until all of the following conditions occur: (1) the CTI server becomes available; (2) the switch being recorded by this channel becomes available; and (3) a configuration option for the channel indicates it is to be driven from an online CTI server and switch.

Degraded Mode

If a record channel is configured to be driven from a CTI source, only CTI MCRs are entered into the database. These CTI MCRs absorb any recorder metadata that intersects with the time ranges of the CTI events. No VOX MCRs are made persistent. If, however, the CRG detects that the CTI Server, switch, or associated communication paths are down, the channel enters Degraded mode. This mode is similar to Initial mode in that VOX MCRs are made persistent when completed. Any CTI MCRs that were left open at the time the CTI Server went down are closed and updated for the last time. The recorder channel will remain in this state until the three conditions indicated in "Initial Mode" are met. Only then will the recorder channel transition into Normal mode.

Normal Mode

Under normal operating procedures in a system with a CTI server and switch online, MCRs are created whenever a VOX or CTI connect event is received and stored in the appropriate list. For each VOX message received, the CTI History List is swept to see if audio metadata can be absorbed by a matching MCR. Any remaining audio data is placed in a VOX MCR. For CTI events involving updates to Recorder Participants, the list of VOX MCRs is swept to see if audio metadata can be absorbed. CTI MCRs are made persistent to the Local Datastore when first created, upon significant update events, and when completed. VOX MCRs are not made persistent to the Local Datastore as they should be completely absorbed by CTI MCRs. There is a configuration parameter that can enable leftover VOX MCRs to be made persistent when they are removed from the VOX MCR history list.

Transitions from Initial/Degraded to Normal Mode

When a CRG channel is in Initial or Degraded mode, VOX MCRs are recorded into the Local Data Store when completed. If notification is received indicating a recorder channel meets the three criteria indicated in "Initial Mode", the channel is set to Normal mode. From this point on, only CTI based MCRs are made persistent and VOX MCRs will be absorbed by the VOX events. Since CTI events represent an accumulated history of a phone call, prior events occurring while the connection between the CRG and CTI Server was lost (or was not yet established) are nonetheless summarized in each update message. The time spans of Recorder Participant(s) are compared to audio data in the VOX MCR list, with any overlaps causing the audio data to be absorbed. In this way, any audio data that occurred while a connection to an external component is temporarily unavailable will still be capable of being correctly associated.

Transitions from Normal to Initial/Degraded Mode

When the CTI server and switch becomes available for driving the call record creation and processing, the CRG channel enters into Normal mode. A heartbeat message is used to periodically update the status of the switch and CTI Server. When the heartbeat is lost or there is a message indicating one of the components has gone down, the recorder channel switches to Degraded mode. The CRG will still create and maintain MCRs in the VOX list and force MCR closure on open CTI MCRs as they pass out of the CTI history buffer. The sweeping action of audio metadata among incomplete CTI MCRs will cease, preventing all future audio data from being absorbed by it. VOX MCRs are made persistent in the database when they leave the history buffer.

US 6,249,570 B1

35

Trunked Radio Mode

In an alternate embodiment of the subject invention, fields in the call record structure are added to support trunking radio. Information contributing to these fields may be obtained from communications with a Motorola SmartZone system. This system uses the Air Traffic Information Access (ATIA) protocol to communicate metadata related to radio activity. The embodiment has a trunking radio server similar to the CTI server that provides an interface between the SmartZone system and the recorders of the preferred system. This server provides the normalization of data and distribution to the correct recorder. There are currently two modes of operation of the Motorola trunking radio system that are discussed below.

Message Trunking

In this mode, when a radio is keyed, it is assigned a particular frequency to communicate on. When the radio is de-keyed, a message timeout timer (2–6 seconds) is started. If another radio in the talk group keys up during this time, the controller uses the same frequency for transmission and resets the timer. The conversation will remain on this frequency until the timer is allowed to expire. During this time, all events that are reported with respect to this conversation will have the same call number associated with them. Therefore, the concept of CTI based call records with many participants has been applied to Message Trunking.

If the timer is allowed to expire, future radio transmissions will be assigned to another frequency and call number. The server needs to detect this occurrence and properly terminate a call record.

Transmission Trunking

Transmission Trunking does not use the holdover timer mechanism used in Message Trunking. When a radio is keyed, it is assigned a particular frequency for transmission. When de-keyed, the channel frequency is immediately freed up for use by another talk group. Therefore, a conversation can take place over many channels without a call number to associate them. The concept of VOX based call records which contain one radio clip per MCR is used in this mode.

Selective Record

There may be certain phone calls involving extension or agents that are not to be recorded. Selective Record is a feature that tells the system to refrain from recording a call while a certain condition exists.

Virtual CRG

MCRs can exist in the subject system's database that have no audio associated with them. These non-audio MCRs can be created due to different features of the subject system. Some customers may require that all CTI data coming from their switch be saved even though they are not recording all extensions or trunk lines. By creating records from the CTI data alone, in the absence of recorded audio, this mode of

36

operation can provide the customer with useful information for statistical analysis or charting purposes. Likewise, records created based upon CTI data alone may provide a useful audit trail to verify the occurrence of certain telephone calls, analyze traffic patterns, or to perform other types of "data mining" operations. In that case, a CRG is associated with the CTI Server mechanism to receive all CTI events that are not matched to a specific recorder. These CTI MCRs are made persistent to the Central Database upon call completion.

Call Record Structure

Call record start and stop events originate from two independent sources: the Recorder and the CTI server. The CRG must perform some method of merging events from these two sources in such a way that the resultant call record contains the best information available. CTI server events are advantageous in that they provide more information than the recorder and can also accurately determine a call record boundary. Recorder based events are a subset of CTI server events and can only distinguish call record boundaries based upon VOX or off/on hook. The recorder has advantages in that since it is in the same box as the CRG, receipt of these events is guaranteed as long as the recorder is running. The main purpose of the assembly process is to leverage the information coming from the CTI server in such a way that the entire phone call is assembled into one Master Call Record (MCR). The structuring of call records is weighed towards trunk side recording with the services of the CTI server driving call record creation. This type of configuration enables the system to summarize phone calls in the most effective manner. The manner in which the structure of the MCR designed to achieve this goal is discussed below.

Master Call Record

The MCR holds information accumulated for all events received necessary for archiving to the local data store. It consists of individual fields that are global to the entire call record as well as lists of specific information. Global information includes identifiers for the call record, the start and stop times of the entire call, the recorder location with respect to the switch, and flags indicating the call record status.

Lists included with each MCR contain the following information: Media File List—List of media filenames that make up the call (e.g., telephone or radio communications); Screen Data Capture File List—List of screen image files associated with audio on this channel; and Participant List—List of participants involved in this call.

The MCR is populated from events received from the CTI Server and Recorders. The following table shows the fields in the MCR, in a preferred embodiment, their data types, description and if they are stored in the database.

Master Call Record structure.

Name	Type (max length)	Archive	Description
m__CallRecID	string¶	Y	Unique ID (UUID) pertaining to entire call (Ctl and Trunk Radio server provides same ID for call parts that are related to the same conversation.)
m__MetaDataSource	BYTE	Y	Indicates the source used to populate call record information. 0 = none 1 = CTI 2 = Trunking Radio
m__bCallComplete	bool	N	Indicates the end of a call. (i.e., there are no more active participants involved)

US 6,249,570 B1

37

38

-continued

<u>Master Call Record structure.</u>			
Name	Type (max length)	Archive	Description
m_bCallHoldoverExceeded	bool	N	If true, MCR has been in a complete state for a time period exceeding the configured Call Holdover time.
m_bMetadataHoldoverExceeded	bool	N	If true, MCR has been inactive for a time period exceeding the configured Metadata holdover period. Used to allow completion of MCRs that haven't been updated for long periods of time possibly because of missed events.
m_bLastUpdate	bool	N	true when the CRG has decided to send the last update of this MCR. Used to prevent any future updates.
m_bDontArchive	bool	N	Indicates whether this call record is to be archived by data store. Certain record features such as selective record may prevent us from storing this call record.
m_CallDirection	BYTE	Y	Indicates call origin Outbound = 0 x 12, Inbound = 0 x 21, Internal = 0 x 11, Unknown = 0 x 44
m_CustomerNumber	string†	Y	Variable length character array dedicated to information the switch may provide with the call. For custom call record support. (e.g., account number)
m_pRecLoc	RecorderLocation	N	Pointer to recorder location descriptor associated with this channel. (see RecorderLocation class)
m_SSFile	list†	Y	List of TimestampedFilename (see below) objects representing Screen Data Capture filename(s) associated with a call record
m_Participants	list†	Y	Array of Callparticipants (see below) describing all participants involved in the call
m_XactionSema	HANDLE	N	Semaphore used to lock this MCR from being modified by any other threads.
m_SemaTimeoutVal	unsigned long	N	Maximum time thread is blocked on m_XactionSema access before returning.
m_bModified	bool	N	Set whenever MCR is changed in a way that requires update to the Local Data Store.
<u>VOX Call Record (Derived)</u>			
m_dwVoxCrNum	DWORD	N	Sequence number of first VOX MCR associated with this CTI MCR (if applicable).
m_bVoxInProgress	bool	N	Indicates this VOX clip is still active (i.e., End time is default time.)
m_CreationTime	time_and_date†	N	Holds time at which the call record was created. Used for debugging purposes to measure how long a call record is alive.
m_CloseTime	time_and_date†	N	Local time at which MCR was marked complete. Used for determining when call record is ready for archive.
m_MediaFiles	list	N	List of TimestampedFilename classes representing multimedia files used to store data with respect to this call record.
m_CtiInfo	CtiInfo	N	Class containing CTI type data associated with call record.
<u>Base Call Record (Derived)</u>			
m_wVersion	WORD	N	Version number of call record.
m_StartTime	time_and_date†	Y	Start time of call record
m_EndTime	time_and_date†	Y	End time of call record

US 6,249,570 B1

39

40

RecorderLocation			SwitchCharacteristics			
m_MetadataServerStatus	BYTE	Indicates the status of the metadata server driving call records for this particular recorder location. This source is in most cases the CTI server but can be other servers such as Trunking Radio Server 0 = "down", 1 = "up"	5	m_bTimeSynced	bool	Indicates if switch is synchronized with the system.
				m_bRealTime	bool	Indicates if switch provides CTI info in realtime (true) or batched and sent periodically (false)
m_SwitchStatus	BYTE	Indicates the status of the telephone switch providing call record information for this particular Recorder Location. 0 = "down", 1 = "up"	10	m_iCmdTimeOffset	int	Value that indicates any known time offset between events received at the switch versus the time the similar signal is received at the recorder. This value will be used to adjust CTI generated timestamps before comparing to recorder events
			15	m_iSwitchTimeOffset	int	For switches that are not time sync'd with the system, this value indicates any known time offset between the switch and the system time. This can be utilized if has some way of updating the time delta between switches and our system on a periodic basis.
m_ExternMetaDataSource	BYTE	Indicates what external source (if any) is contributing call record meta data for this channel. 0 = None (recorder only) 2 = CTI server	20			
m_ChanID	ChannelIdentifier	Class identifying recorder channel.	25	CTIInfo		
m_SwitchID	Switch Identifier	Class identifying switch connection point.	30	RingLength	WORD	Time (in sec) between first ring signal and off hook.
m_SwitchChars	SwitchCharacteristics	Class identifying characteristics of switch needed by CRG.	35	DTMFCode	string† (50)	DTMF codes entered during conversation
			TimeStampedFilename			
			40	Name	Type	Description
				m_AFStartTime	Time_and_date†	Start time of audio file
				m_StartTime	Time_and_date†	Start time of interest
				m_EndTime	Time_and_date†	End time of interest
			45	m_SegStartTime	Time_and_date†	Start time of segment inside file absorbed by this MCR.
				m_SegEndTime	Time_and_date†	End time of segment inside file absorbed by this MCR.
				m_PathName	string† (36)	Path describing the Voice Server and directory location where the audio files are located.
			50	m_File Name	string† (36)	GUID-based name that uniquely identifies a specific audio segment's recording file.
				m_wFileType	WORD	bitmap indicating types of media associated with MCR.
			55		bit	Data
					0	Audio Present
					2	FAX Present
					3	Video Present
					3	Screen Capture data Present
			60	m_wFileFormat	WORD	Recording format of media data, as defined by Microsoft Corporation's multimedia description file "mmreg.h"
				m_bNew	bool	Used by local data store to indicate whether this record should be inserted (true) or updated (false) into the database.
			65			
SwitchIdentifier						
m_SwitchNum	WORD	Number identifying switch attached to switch. (Valid only if not equal to -1)				
m_wTrunkID	WORD	Identifies time slot of digital line (T1 or E1) of interest. (Valid only if TrunkID is not equal to -1)				
m_dwVirtualChannel	DWORD	Extension number (Valid only if m_wTrunkID equals -1)				
m_Extension	string† (6)					
ChannelIdentifier						
m_wNode	WORD	Unique number used to distinguish between multiple Voice Servers.				
m_wChannel	WORD	Unique number used to distinguish between multiple recording input channels within a Voice Server.				
m_bSignalSupport	bool	Indicates if hardware associated with this channel supports on/off hook signaling.				

US 6,249,570 B1

41

-continued

TimeStampedFilename		
Name	Type	Description
m_bDiscard	bool	If true, don't allow playback or archiving of this media. Used for the Selective Record feature.
m_dwVoxCrNum	DWORD	Sequence number corresponding to VOX call record that provided this media.
m_iAssocPart	int	Index of Recorder Participant in the Participant list causing this media file to be associated with this MCR.

CallParticipant		
Name	Type	Description
m_AgentID	string† (24)	Registered ID of agent at extension (CTI) or Radio Alias (Trunking Radio).
m_Number	string† (24)	Full telephone number of the participant (i.e., ANI, DNIS)
m_Console	string† (10)	Seating position of participant that can consist of one or more stations (CTI) or Talkgroup ID (Trunking Radio).
m_Station	string† (10)	Unique telephone set. Possibly with multiple extensions
m_LocRef	BYTE	Describes the location of participant with respect to the switch. (1=internal, 2=external, 3=unknown)
m_SwitchLoc	SwitchIdentifier	Class identifying the position of a participant relative to the telephone switch.
m_StartTime	Time_and_date†	Time participant joined the call
m_EndTime	Time_and_date†	Time participant left the call
m_ConnectReason	BYTE	How participant joined the call NotConnected = 0, NormalStart = 1, ConferenceAdd = 2, TransferRecv = 3, UnknownConnect = 9
m_DisconnectReason	BYTE	How participant left the call NotDisconnected = 0, NormalEnd = 1, ConferenceDrop = 2, TransferAway = 3, OtherPartyHangup = 4, UnknownConnect = 9
Changed	Bool	Indicates if recent change in CTI message. (not archived)
Trunking Radio Only Information		
SourceSiteID	BYTE	Site number that is currently sourcing audio on active call.
ZoneID	BYTE	Zone at which participant is currently located.
CIUNumber	BYTE	Console Interface Unit. Translates 12kbit into clear audio & vice versa.
CDLNumber	BYTE	Channel associated with CIU

42

-continued

CallParticipant		
Name	Type	Description
DIUNumber	BYTE	Digital Interface Unit. Translates ASTRO clear secure data into analog audio & vice versa.
DBLNumber	BYTE	Channel associated with DIU

† -Objectspace data types

Unused string fields are null. Unused number fields are set to zero.

15 The version number is used to indicate the structure of data contained within the call record. In order to maintain compatibility with future versions, changes to call record structures will be performed in an additive nature. That is, current members of the call record will not change in position, size, or meaning.

Each call record will contain a list to store participant information. There will be at least two participants in a call record; the calling and called parties. Any additional connections that are conferenced in or transferred to are appended to the end of this list.

Only one active VOX and CTI based Master Call Record is allowed per recording input channel at any given time. CRG Software Architecture

FIG. 18 shows the processing threads and data structures that comprise the CRG module in a preferred embodiment.

Event Processing

When the CRG is created and initialized, three threads are created. These threads are the CRG Event Processor thread 1810, Façade thread (The terms "façade," "facade," and "fascade" are used interchangeably in this disclosure) 1812 and Local Data Store thread 1816. Additionally, three message queues are created and are known as the Recorder 1824, Façade 1832, and Data Store 1844 queues, respectively. These queues enable the processing of various input messages in a de-coupled fashion within the CRG, so that any delay in one area of communications does not affect the processing of another area. Each thread is described below.

Event Processor Thread

The Event Processor is the primary thread of the CRG module. Its responsibilities include reading any messages placed in the Recorder 1824 and Façade 1832 queues. The processing activities that occur in response to these messages cause updates to be made to call records belonging to one of the recording input channels 1856. If these changes cause a call record to be completed, a message is sent to the Date Store queue 1844 requesting that the call record be made persistent in the local database. This thread is also responsible for processing state change messages, that cause memory resident structures to be refreshed or to shut down the CRG module.

Façade Thread

The Façade thread handles messages that come from outside the Voice Server. Its primary function is to look for messages placed in the CRG's external Microsoft Message Queue (MSMQ) 1864 where events may arrive from other components within the overall subject system. Upon receipt of a message, the Façade thread reads the message, translates it into an appropriate format for the CRG's internal data structures, and places the translated copy in the Façade Queue 1832. This thread is known as the Façade, because it manages the external interactions of the CRG with the other components within the subject system.

US 6,249,570 B1

43

Local Data Store Thread

The Local Data Store thread 1816 processes requests from the CRG Event Processor thread 1810. The primary purpose of the Local Data Store thread 1816 is to take internal Master Call Record (MCR) structures and translate their contents into structures compatible with database technologies, such as Microsoft SQL Server, or comparable types of storage means. These resultant structures are stored within the database in order to make the call record persistent.

Characteristics of some switches mandate that the CRG be able to handle CTI events that are not real-time. Some switches batch events and send them out periodically. CRG configuration settings that limit the history list by time must be set long enough to accommodate the switch characteristics. Therefore, call records that are generated between switch reports (via recorder events) will not be finalized until a configurable time period (window) after which the call record terminated. This window (CallHoldoverPeriod) needs to be set to a minimum of the period of time between switch reports. Once a call record leaves this time window, it is marked as read-only and committed to the local data store.

A situation that must be dealt with is when the telephone switch is not time synchronized with the rest of the system. To facilitate the merger of recorder and switch events effectively in non-time-synchronized systems, alternate embodiments of the subject system are described.

One alternate embodiment of the subject system has a mechanism that synchronizes the clocks in the system (manually or automatically) on a periodic basis. This must guarantee time skews of less than some small and known quantity. A second embodiment has a mechanism for measuring the time delta between the switch and the subject system. This value is updated periodically and used by the CRG during the merging process. A third embodiment implements a combination of the first two.

During the call record merging process, a global time delta is used to adjust switch event time stamps before comparing to existing call record data.

The following paragraphs define the types of events the CRG is designed to accept and process. These events may cause the CRG to initialize, process metadata into call records, or prepare the system for shutdown.

The Master Controller (a sub-component of the present system's Scheduling & Control Services) supplies system events. The Master Controller notifies the CRG of system related changes such as configuration changes, CTI server status and system shutdown events. The CRG changes its behavior based upon events received from the Master Controller.

System Events

The CRG provides an interface that allows the client application to control its states of operation. This is accomplished with an interface class that is used by most system components in the subject system. The interface is named IProcCtrl and supports the following methods: Initialize(); Start(); Stop(); Pause(); Resume(); Ping(); and Shutdown().

In addition to these methods, the CRG supports two event messages that inform it of status changes that are needed to either update its memory resident configuration information or change its mode of operation. These methods are CtiStatus and AgentExtensionStatus. Each method is described in the following paragraphs.

Initialization Event

This method is the first method that should be called after the CRG has been created. When the CRG object is created,

44

it retrieves configuration information from the subject system's database. This information describes the number of channels in the recorder, the switch location where each channel is connected, any fixed associations of telephone extensions or agent identifiers. Also included are parameters that determine the behavior of the CRG. Threads are spawned to handle the processing of CRG events, communicating with external metadata contributors, and processing information into the Call Records tables. These threads are created in a suspended state and require the Start or Resume commands to begin processing activity.

Start Event

This method should be called after the Initialization event. It resumes all threads of the CRG enabling it to process incoming events.

Pause Event

This method suspends all threads of the CRG.

Resume Event

This method is called after the Pause command to enable all CRG threads to continue processing.

Ping Event

This method is used by client applications to test the connection to the CRG. The method simply returns a positive acknowledgment to let the client know that the CRG is still running.

Shutdown Event

This method notifies the CRG when the subject system is shutting down so that it can cleanly terminate itself. The shutdown event supports a single parameter (ShutdownMode) that indicates how it should shutdown.

If the ShutdownMode is specified as "Normal", all pending events read from the input event queues and processed into the call records, any open call records remaining are closed at the current time and written to the database.

If the ShutdownMode is "Immediate", input event queues are cleared without processing into call records, open call records are closed and written to the database.

Once these actions are completed, the CRG threads terminate. At this point, it is now safe for the client application to release the resources of the CRG.

Stop Event

This method is implemented for consistency with the common interface of IProcCtrl. The CRG has no purpose for this method and just returns a positive acknowledgment.

CtiStatus Event

This event informs the CRG of the operational status of the CTI server that is providing it with telephony metadata needed for CTI call record generation. The Scheduler component of the subject system is responsible for maintaining a heartbeat with the CTI server to detect when connection has been lost. Any changes in CTI server status result in a CtiStatus message directed at the CRG.

This message contains one parameter that indicates the new state of the CTI Server. If the parameter indicates that a CTI Server has become operational, recording input channels associated with the CTI Server change from "Degraded" mode of operation of "Normal" mode. If the parameter indicates that the CTI Server is not operational, recording input channels associated with the CTI Server change from "Normal" mode of operation to "Degraded" mode.

AgentExtensionStatus Event

This event indicates that a change in one of the Agent or Extension tables has occurred. Since the CRG uses these tables to associate with recorder channels, the memory resident version must be updated. Therefore, this event causes the CRG to read these tables and update its memory resident copy.

US 6,249,570 B1

45

Call Record Events

When a call record event is received, the message is interpreted to determine which recording input channel may be affected. Any filtering necessary on a per channel basis is performed at this stage. Call record events are then dispatched to the appropriate Call Record Channel Manager. There is a separate call record channel manager, which is a software sub-component of the CRG, for each recording input channel in a Voice Server. There are three messages that directly contribute to the creation and completion of call records. One comes from the CTI Server in the form of a CTI Event. The other two originate from the recorder and are the VoxSummary and VoxDisconnect messages. Each message is described in detail below.

CTI Event

The CTI Event is a message originating from the CTI Server software module that processes the information received from the telephone switch. The message details each participant involved with the phone call as well as information global to the call such as ring duration and DTMF codes. A CTI event message is sent to the CRG whenever a change in participant status occurs as well as when new ones enter the call. The messages are cumulative in that all information of the previous messages is contained in the new one with any additions included. This makes for a more robust system in cases where one of the messages is lost.

The pseudo code for processing a CTI event is shown below:

Pseudo Code for CTI Event

```
// - - - CTI Event (BEGIN) - - -
```

```
// Don't process CTI events if we're not in correct mode
```

```
Is this recorder channel configured to receive CTI event data?
```

```
{// Yes
```

```
Does this event match an MCR in my CTI History list?
```

```
{// Yes
```

```
Update MCR participants with matching one in CTI event
```

```
Add any new participants to MCR.
```

```
UpdateMediaFiles( ) (see pseudo code)
```

```
// End—Does this event match an MCR in my history list?
```

```
Otherwise
```

```
{
```

```
Create new MCR
```

```
Initialize MCR Start time from Oldest Participant Start time in event
```

```
Copy participants from event to message to MCR.
```

```
// Now that we've updated the participants, see if
```

```
// we need to change media file associations.
```

```
UpdateMediaFiles( ) (see pseudo code)
```

```
Insert new MCR Into Cti MCR history list.
```

```
}
```

```
Are there any participants still active?
```

```
Mark MCR as active
```

```
Otherwise
```

```
Mark MCR as complete
```

```
// End—Is this recorder channel configured to receive CTI event data?
```

```
// - - - CTI Event (END) - - -
```

```
// - - - UpdateMediaFiles (BEGIN) - - - for each Recorder Participant in the MCR
```

46

```
{
```

```
Is this not a new Recorder Participant?
```

```
{
```

```
//This participants start and/or end time may have been adjusted.
```

```
//See if audio previously absorbed by it has to be returned to the VOX history list
```

```
FindGiveBackMediaFiles( ) (see pseudo code below)
```

```
}
```

```
for each MCR in VOX History list with a time range that overlaps with this recorder participant
```

```
{
```

```
for each media file in this VOX MCR that's timespan overlaps with this recorder participant
```

```
{
```

```
CheckAndApplyMediaFile( ) (see pseudo code)
```

```
}End—for each media file in this VOX MCR that's timespan overlaps with this recorder
```

```
participant
```

```
Did we consume all audio in this VOX MCR?
```

```
Remove VOX MCR from History list and delete it.
```

```
}End—for each MCR in VOX History list that's time overlaps with this recorder participant
```

```
}End—for each Recorder Participants in the MCR
```

```
GiveBackAudio( ) (see pseudo code)
```

```
// - - - UpdateMediaFiles (END) - - -
```

```
// - - - FindGiveBackMediaFiles (BEGIN) - - - for each media file associated with the given CTI MCR
```

```
{
```

```
Was this media file contributed from the given recorder participant?
```

```
{//Yes
```

```
if media file lies completely outside recorder participant timespan?
```

```
|<- - - Participant Timespan - - - ->|
```

```
|<- - - Media File timespan - - - ->|
```

```
Move this entire media file to the Giveback list
```

```
Otherwise, If media file start time is before the recorder participants start time?
```

```
|<- - - Participant Timespan - - - ->|
```

```
|<- - - Media File Timespan - - - ->|
```

```
Make a copy of this media file and set its end time to the participants start time.
```

```
Add media file to giveback list.
```

```
Set original media files start time to that of recorder participant.
```

```
Otherwise, if media file end time is after the recorder participants end time?
```

```
|<- - - Participant Timespan - - - ->|
```

```
|<- - - Media File Timespan - - - ->|
```

```
Make a copy of this media file set its start time to the participants end time.
```

```
Add media file to giveback list.
```

```
Set original media files end time to that of recorder participant.
```

```
}End—for each media file associated with the given CTI MCR
```

```
// - - - FindGiveBackMediaFiles (END) - - -
```

```
// - - - GivebackAudio (BEGIN) - - -
```

```
// Sweep through VOX MCRs re-populating any giveback audio for all audio portions in given back list
```

US 6,249,570 B1

47

```

{
if we find the VOX MCR this audio originally came from?
{ // Yes
  Attempt to merge the give back media file with an existing
  VOX MCR media file
  that's start or end time is adjacent to this ones.
  Otherwise, associate this media file with the VOX MCR.
} End—if we find the VOX MCR this audio originally came
from?
Otherwise
{
  // Original VOX MCR containing this audio file doesn't
  exist anymore.
  Create a new MCR.
  Associate the giveback media file with the new MCR
  Insert MCR into VOX History list
}
} End—for all audio portions given back
// - - - GivebackAudio (END) - - -

// - - - CheckAndApplyMediaFile (Begin)
Does Recorder Participant span the entire media file?
|<- - - Participant Timespan - - ->|
|<- - - Media File timespan - - ->|
Move media file from VOX MCR to Cti MCR.

Otherwise, Does Recorder Participant overlap with media
file start time?
|<- - - Participant Timespan - - ->|
|<- - - Media File timespan - - ->|
Make a copy of this media file and set its end time to the
participants end time and associate with recorder partici-
pants MCR.
Set the original media files start time to the recorder par-
ticipants end time.

Otherwise, Does Recorder Participant overlap with media
files end time?
|<- - - Participant Timespan - - ->|
|<- - - Media File timespan - - ->|
Make a copy of this media file and set its start time to the
participants start time and Make another copy of this
media file and set its start time to the participants end time
and associate with VOX MCR. Set the original media files
end time to the recorder participants start time.
// - - - CheckAndApplyMediaFile (End)
VOX Summary Event
The VOX Summary Event is a message originating from
the recorder associated with this CRG. It can be used in one
of two ways.
The primary use of this message is to indicate the start of
audio activity in real-time. When used in this mode, the
VOXSummary command indicates the beginning of audio
activity. But since the activity is not complete, the end time
is set to indicate that the VOX segment is incomplete. The
end time of incomplete media file is also set in this way. In
this case, a VOX Disconnect message is required to com-
plete the end times.
The second mode is used to indicate a history of audio
activity. The VOX Summary start and end times reflect the
period of time covered by all accompanying media files. The
media files also have there respective start and end times
filled in. This message is complete and thus requires no
follow up messages. The VOXSummary message is shown
below.

```

48

VOX Summary Message Format

Field Name	Description
Channel	Recorder channel of audio activity
VOXCrNum	Sequence number used to correlate related VOX events.
StartTime	time at which audio activity first started
EndTime	Time at which last audio activity ended.
Media Files	list of multimedia filenames used to store data with respect to this call record. (see below for details)
RingLength	Time from start of ring to off hook (in sec)
DtmfCodes	String of DTMF codes detected during VOXSummary period
ConnectReason	Indication of why VOX segment was started
DisconnectReason	Indication of why VOX segment was terminated

Media File Structure

Field Name	Description
FileStartTime	Time corresponding to first byte of audio data in a file.
StartTime	Time corresponding to first byte of audio at which activity occurred
EndTime	Time corresponding to last byte of audio at which activity occurred
FileName	String containing name of audio file.
PathName	String describing the location of audio file.
iAssocPart	Used by the CRG to indicate with which Recorder Participant this audio segment is associated, when it is part of a CTI-based MCR.
dwVOXCrNum	Used by CRG to indicate which MCR in the VOX History list this audio segment originated.

The pseudo code for processing a VOX Summary event is shown below.

```

// - - - VOXSummary (BEGIN) - - -
Is this recorder channel configured to receive CTI event
data?

```

```

{ // Yes
  // Attempt to merge media files in message with CTI based
  MCRs
  for each CTI MCR in History list
  {
    If any of the given media files fall inside the timespan
    of the Cti MCR?
    { // Yes
      // Merge media files with overlapping recorder par-
      ticipants in CTI MCR
      for each given media file in VOX Summary message
      for each recorder participant in the given CTI MCR
      {
        CheckAndApplyMediaFile ( ) (see psuedo code)
      } End—for each recorder participant in the given
      CTI MCR
    } End—for each given media file
  } End—If any of the given media files fall inside the
  timespan of the Cti MCR?
} End—for each CTI MCR in History list
Remove media files from VOX Summary message that are
completely consumed

```

```

Any unabsorbed audio remaining in message?

```

```

{ // Yes

```


US 6,249,570 B1

49

Create MCR for remainder of audio.
 Insert MCR into VOX History List
 }
 // - - - VOXSummary (END) - - -
 VOX Disconnect Event
 The VOX Disconnect Event is a message originating from the recorder associated with this CRG. It is used to terminate a VOX segment that has been started by a real-time VOX-Summary message.

The VOXDisconnect message is shown below.

VOX Disconnect Message Format	
Field Name	Description
Channel VOXCrNum	Recorder channel of audio activity Sequence number used to correlate related VOX events.
Time	End time of the VOX segment. Also indicates the end time of open media file.
DisconnectReason	Indication of why VOX segment was terminated

The pseudo code for processing a VOX Disconnect event is shown below.

```
// - - - VOXDisconnect (BEGIN) - - -
Is there a MCR in VOX History list with the same sequence number?
{ // Yes
  // Close and update all media files in both VOX and
  // MCR list related to this one
  Close Active media file in VOX MCR at given message time
  UpdateFromMediaFile( )
  // Update any CtiMCRs that absorbed the audio file
  // closed.
  for each MCR in CTI History list
  {
    // Attempt to merge audio with MCR.
    // Look for matches with audio filenames.
    for each media file in Cti MCR contributed by this
    VOX clip
    {
      Close media file at given message time

      // Now that we've closed it, does this media file still
      // belong with this CtiMCR?
      Does media file still fall in time span of MCR?
      { // Yes
        CheckAndApplyMediaFile ( ) (see pseudo code)
      }
      Otherwise
      {
        Remove media file from MCR list and discard
      }
    }
  }
  End—for each media file in MCR contributed by this
  VOX clip
}
Close VOX MCR and mark as complete
}
// - - - VOXDisconnect (END) - - -
// - - - UpdateFromMediaFile (BEGIN) - - -
// Look for matches with audio filenames for each media
// file in MCR contributed by this VOX clip
{
  Close media file at given message time
```

50

```
// Now that we've closed it, does this media file still
// belong with this CtiMCR?
Does media file still fall in timespan of MCR?
{ // No
  CheckAndApplyMediaFile ( ) (see pseudo code)
}
Otherwise
  Remove media file from MCR list and discard
}
}
End—for each media file in MCR contributed by this
VOX clip
// - - - UpdateFromMediaFile (END) - - -
Data Events
Data events are appended to the currently open associated
call record. For CTI data events, this pertains to a currently
open MCR based upon CTI connect events and containing
a matching call record ID. For VOX data events, the
currently open VOX call record is affected. If an open call
record doesn't exist, an error condition is reported.
Correction Events
Correction events exist to remove a previous alteration to
a call record after it has already been populated. One reason
for such an event is to support selective record. An audio file
that cannot be recorded due to customer or legal reasons
might need to be removed from the call record or the entire
call record might need to be deleted. The VOX event for a
filename might have already been processed into a call
record before the selective record mechanism has deter-
mined it not to be recorded.
Selective Record (Exclusion)
Selective Record is an important feature of the subject
system, imposed by customer requirements. If the customer
does not want certain participants recorded when they
become involved in a recorded call, the CRG must exclude
any audio associated with the call record for that partici-
pants' time of involvement. Implementing this feature is
complicated by the varying characteristics of customer
switches. If the telephone switch environments report events
in real-time, recording of media can be prevented by turning
the recording input channel off during the selective record
participants' time of involvement. However, what happens
when events are not reported in real time from the switch?
The answer lies in the sweeping action of the CRG previ-
ously discussed for recorder participants.
The CTI Event message is routed through the Scheduler,
and is altered by the Scheduler to indicate which participants
re recorder participants as well as which ones are selective
record participants. Recorder participants trigger the CRG to
sweep any audio from VOX MCRs that overlap in time.
When the CRG detects an overlap between recorder partici-
pant and selective record participant times, the audio that is
swept into the CTI MCR for this overlap period is discarded.
This causes the audio to be removed from both VOX and
CTI MCRs, which prevents any chance of the audio being
made available for playback or archive.
Selective Record Event
The Selective Record command is an event originating
from the Scheduler. It identifies either a participant that is
not to be recorded or that an entire call record should not be
recorded. In one embodiment the system is capable of
handling recording exceptions based upon information
obtained from the CTI data. Criteria for selective record
processing are discussed below.
Selective Record feature can take on two meanings. In
one instance, a customer may want to record all telephony
events except for ones that meet specific criteria. In a second
instance, a customer may only want to record calls that meet
certain criteria.
```

US 6,249,570 B1

51

Since selective recording can possibly be triggered from multiple sources, in a preferred embodiment this decision process is located in the Master Controller, a sub-component of the subject system's Scheduling & Control Services.

Suggested reasons for not recording all or parts of a call are based upon the following examples of CTI event data.

Event Data	Explanation	Results
Agent exclusion based upon participants AgentID	Supervisor involved calls not to be included	Delete audio for agent's participation during the call, and the associated references in the MCR.
Exclusion based upon Extension or fully qualified phone Number of participant.	CEO involved calls not to be recorded. (whether at office or at home)	Delete audio for agent's participation during the call, and the associated references in the MCR.
Combination of AgentID of one participant and fully qualified phone number of another participant	Prisoner calls his lawyer.	Delete all audio as well as the entire call record.

Based upon these conditions and any future rules established inside the Master Controller (MC), exclusion can take place on audio recorded during a target participant's time of involvement or over the entire call record.

The chain of events involved in Selective Record (Call Exclusion) is as follows:

1. Recorder detects presence of audio and records to audio buffer.
2. Recorder sends VOX events to CRG indicating presence of audio.
3. CRG creates new call record based upon VOX event.
4. The CTI server sends call events to CRG and MC.
5. CRG associates CTI event data with VOX based call record.
6. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a Selective Record (exclusion) command is sent to both Recorder and CRG indicating the start of the selective record interval.
7. Recorder deletes audio indicated in selective record message and continues to suppress recording until instructed otherwise.
8. CRG alters the call record to eliminate details of participant or deletes the call record.
9. Upon completion of the call, the CTI Server sends call events to the CRG and MC.
10. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a selective record (exclusion) is sent to the Recorder indicating the end of the selective record interval.
11. The Recorder resumes its normal mode of audio recording.

Selective Record (Call Inclusion)

1. The CTI server sends call events to CRG and MC. CRG creates MCR and populates with events. Since default is set not to record, the flag `m_bDontArchive` is set to prevent the local data store from writing it to the database.
2. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a Selective Record (inclusion) command is sent to both Recorder and CRG indicating the start of the selective record interval. CRG sets `m_bDontArchive` to false and immediately instructs local data store to archive.

52

3. Recorder detects presence of audio and records to audio buffer.
 4. Recorder sends history of VOX events to CRG in a VoxSummary message.
 5. CRG creates new call record based upon VOX event.
 6. CRG associates CTI event data with VOX based call record.
 7. Upon completion of the call, the CTI Server sends call events to the CRG and MC.
 8. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a selective record (inclusion) command is sent to the Recorder indicating the end of the selective record interval.
 9. The Recorder resumes its normal mode of suppressing the audio recording.
- The format of the Recorder's Selective Record command is shown below.

Name	Type	Description
StartTime	time_and_date	Start Time of recording interval
EndTime	Time_and_date	End Time of recording interval
bRecordAudio	bool	If true, record audio during the indicated interval. If false, suppress any audio recording during the indicated interval.

Since the recorder has no knowledge of participants or call record boundaries, the MC needs to inform the recorder when to start a selective record interval and when to stop. The boolean `bRecordAudio` signifies what action should be taken during this interval.

When an event occurs that triggers the start of a selective record interval, the Recorder's selective record command informs the recorder of the interval start. The End time is most likely not known at this point so it is set to some invalid value in order to indicate that audio should be recorded (or suppressed) for an indefinite period until a subsequent command is received.

When an event occurs that triggers the end of a selective record interval, the Recorder's selective record command informs the recorder of the interval end. The End time indicates when the selective record interval is complete. The recorder returns to its normal recording mode based upon its original configuration.

Any selected audio committed to file needs to be removed from the file and replaced with a silence entry for that period.

The format of the CRG selective record command is shown below.

Name	Type	Description
MCR Number	UUID	MCR affected by this selective record command
Participant Index	UINT	Index of participant in MCR not to be recorded. (if Reason=1)
Reason	BYTE	1 =Participant, 2=Entire Call

For the CRG, only a single event that indicates what is selectively recorded is needed. If the Reason code indicates that the entire call record is to be deleted, the CRG will mark the call record such that it is removed from the database if

US 6,249,570 B1

53

it has already been written or not logged in the first place. If selective record affects a specific participant, the call record can either be left unmodified (since the recorder has already handled deletion of audio) or the participant can be over-written to remove his/her details.

The system configuration can be adjusted so that the CRG will operate in either fashion, depending on whether removing the audio alone is sufficient for the desired application of the system, or if the metadata must also be removed to eliminate the records of telephone numbers dialed, etc.

CRG Software Implementation

In the preferred embodiment of the subject system, the CRG is implemented as an in-process COM DLL that is associated with the Audio Recorder process, and therefore these two components reside together upon the Voice Server. COM, here, is Common Object Model, a distributed computing architecture designed by Microsoft Corporation to facilitate cooperative processing among software elements on a LAN. DLL is Dynamic Link Library, a means whereby executable code can be encapsulated in a package that can be loaded upon demand and shared by several programs, rather than being packaged as a separate, isolated executable program. The Audio Recorder process is responsible for creating the CRG COM object as well as starting and stopping the CRG subsystem. The Data Store module that interfaces with the CRG is a statically linked DLL.

Class Design

FIG. 19 illustrates the class diagram of the Call Record Generator. The CRG module is itself comprised of a plurality of modules, as shown in the figure, and explained below.

CallRecordEvent Processor—the CallRecordEventProcessor class 1912 is the main class of the CRG. It is instantiated during the Initialize method call of the CRG interface. It is responsible for allocating the rest of the CRG objects. On instantiation, it acquires the channel count for the recorder (currently limited to 128) and instantiates a

54

group a classes for each recording input channel. These classes include a CallRecordChannelManager 1916 and RecorderLocation 1920 for each channel. The CallRecordEventProcessor 1912 creates the Recorder 1924 and Façade 1928 Event input queues. Reading and processing of configuration information from the subject system's database takes place in the CallRecordEventProcessor 1912. Events received that cause a change in configuration are processed there.

10 CallRecordChannelManager—This class manages the call records for a specific recording input channel. It is responsible for creating, populating, and closing call records with event information received from the CRG event processor. If event information is deemed as significant, the CallRecordChannelManager 1916 will send an event to the DataStoreEventQueue 1932 in order for the update to be reflected in the local data store.

MasterCallRecord—This class 1936 holds information that is global to an entire call. Global information includes identifiers for the call record, the start and stop times of the entire call, the recorder location with respect to the switch, and flags indicating the call record status. It also contains a list of the participants within a call, based upon information supplied by CTI events. It acts as a centralized point of control for merging call record information for a given telephone call.

VoxCallRecord—This class 1940 is a superclass of the MasterCallRecord class 1936. It contains information dealing with events provided by the recorder. It holds the details of a call, such as the start/stop times, media filenames and other data that can be supplied by the recorder.

RecorderLocation—This class 1920 holds the information relating a logical device on a telephony switch with a specific Voice Server and recording input channel.

35 The following table indicates configuration information needed by the CRG at runtime.

Configuration Field	Type	Acceptable Values	Default	Description
sSysTimeCoupling	String	"TIGHT", "LOOSE"	"LOOSE"	Indicates how time-based recorder and CTI events are compared to determine a match. TIGHT - Recorder times must fit entirely inside CTI times for a positive result. LOOSE - Recorder times need to overlap with CTI times for a positive result.
nCompleteCallHoldOver Period	DWORD	0 . . . 42949672 96 (in sec)	90 - For realtime CTI. (Much larger if non- realtime CTI)	Maximum number of seconds that a completed call record is kept in the history list. This holdover allows events coming from different sources to affect the call record before it is made persistent. After this holdover period expires, no more events can update the call record.
nActiveCallHoldover Period	DWORD	0 . . . 42949672 96 (in sec)	86400 (24 hours)	The maximum number of seconds a call is allowed to exist before being forcibly closed. This is used as a safeguard against missing CTI or Recorder events that would normally end a call record.
nMCRMaxSize	WORD	0 . . . 65535 (in entries)	100	Maximum number of entries allowed in the MasterCall Record history list

US 6,249,570 B1

55

56

-continued

Configuration Field	Type	Acceptable Values	Default	Description
nSystemSkew	WORD	0 . . . 65535 (in sec)	0	A known, fixed difference (in seconds) that specifies the skew between a Recorder clock and a PBX clock. Used to adjust incoming CTI event times before processing and comparing with Recorder event times.
ynCTIDataFromRecorder	bool	1 = yes 0 = no	yes	Identifies, in cases where Recorder and CTI information overlaps, which source is preferred to populate the call records
nSaveVoxClipsLonger-ThanSeconds	WORD	0 . . . 65535	6	This setting is used to avoid creating VOX based call records from noise on recording input channels. It directs the CRG to discard any VOX clips that do not exceed the specified number of seconds in duration.

Stream Control Manager

As noted above, in a preferred embodiment, the system of the present invention taps into activity on a PBX (Private Branch Exchange) by intercepting audio on either the trunk or extension side of a phone call. The tapped audio is then redirected as input to a channel on a DSP (Digital Signal Processor) based voice processing board, which in turn is digitized and stored into program-addressable buffers. The recorded audio is then combined with descriptive information ("metadata") obtained through a Computer Telephony Integration (CTI) communications link with the PBX and stored as a single manageable unit ("Voicedata") to facilitate its subsequent search and retrieval.

The preferred embodiment leverages Computer Telephony Integration, to supplement the recorded audio data. As discussed above, CTI is provided through a data link from specific telephone switching equipment located at the customer site, which is then input to the recording system's CTI Server. Supplied data includes such items as telephone numbers of involved parties, caller ID/ANI information, DNIS information, and agent ID numbers. The CTI Server performs the task of analyzing and reorganizing data from both the real-time and SMDR (asynchronous) links, and passing the results onwards to the remainder of the recording system for further processing.

A module called the "Call Record Generator," or CRG, discussed above, is then responsible for collecting data from the CTI Server, creating 'master call records' and attempting to match those records with existing recorded audio data. If the CRG receives CTI information indicating that audio data recorded on two Voice Servers is related (for example, due to a transferred call), records will be generated for each portion with a common call record ID. This ID can later be used to query for all the pieces (or "segments") comprising the complete call. In addition, each segment will indicate the Voice Server which contains that piece of the call.

During playback, the User Workstation's player module connects to a program located on a Voice Server called the Playback Server, or PBServer. The machine name of the particular Voice Server with which a communications session should be established, stored by the CRG in the call record table of the Voicedata storage module, is passed into the player module after being extracted by the User Workstation's call record browser. A call record playback request is then submitted, which causes the PBServer to query for a specific call record's audio files located on that physical

machine, open them, and prepare to stream the audio upon buffer requests back to the client. If successful, a series of requests is then issued from the client, each of which will obtain just enough audio to play to a waveOut device while maintaining a safety net of extra audio in case of network delays. Upon a request to "Move" within the scope of a call record, the PBServer will reposition its read pointer to the desired location and then begin passing back buffers from that point. This series of Request and Move commands will continue until the user chooses to end the session by shutting down the client-side audio player.

When a call is transferred between locations, it is possible that the call may span multiple Voice Servers, since the extensions or trunks involved may be monitored by different recorders. If this is the case, the audio data is spread out between playback servers, and it must be properly pieced back together to reconstruct the complete call for a playback client.

There are several possible solutions to the problem. First of all, one could choose one central server and copy in all data from the involved servers. This is as slow as copying the files locally to the client, but it at least consolidates the data to one location for the playback server to operate on. Assuming that this method is chosen, however, several new problems arise. First is the issue of drive space: depending on the number of transfers and recorders involved with a call record, the central playback server could end up suddenly storing a large number of files. This is multiplied by the total number of clients requesting playback sessions. Soon enough, a large amount of unpredictable space is being allocated and freed without any reasonable way of estimating the space necessary to service all requests. Similarly, the processor and memory load on this server is taking the brunt of being used for every playback request, since even normal, single recorder playback sessions would be routed through this one machine.

Another solution would be to have the central playback server run some intermediate process that would stream all of the data from the multiple servers back to each client, like a "funnel." This would avoid the copying and drive space issues, but there are still two problems. First, the centralizing of this server once again puts the entire load on a single machine. But more importantly, if multiple streams are being funneled through this one location, the server would somehow need to organize the streams so that during playback, they appear to be arranged in the proper order.

US 6,249,570 B1

57

The Stream Control Manager (SCM) used in accordance with a preferred embodiment is the result of addressing the issues referred to in the second solution discussed above. With regard to the resource issue, the solution was to simply move the "funneling" module from one central server to the client side. In this way, servers are still providing the actual requested data, but it becomes the client side's responsibility to bring the data together. Yet the SCM remains a separate, COM-based module so encapsulation is still maintained (a client application is not hard-wired directly into the SCM code). This was intentional since other system modules in alternate embodiments of the system need to reuse the SCM to gather playback data (e.g., for phone handset playback support instead of LAN playback support) or to gather audio from a multitude of Voice Servers for long-term offline storage on DAT or DVD media.

The process of stream management begins when the SCM is sent a list of segments which comprise the entire call. Each segment includes the machine name of the Voice Server, the segment's start time, duration, channel ID, and an event callback routine provided by the client which serves as a destination for the final organized data.

Once this list is received and stored as a vector (array), the SCM proceeds to try connecting to all servers required to play back this call. The connection, if successfully established, is associated with its respective segment via a pointer in the segment entry. The connection is also added to an array so that if a subsequent segment's server is the same as an earlier segment, the connection can be reused. This may occur if a call transfers away to a line monitored by a second recorder and is later transferred back again to the original line. If the process cannot complete successfully (i.e., if a Voice Server is malfunctioning), playback is aborted to avoid skipping over any necessary data.

Next, the SCM goes through its list of segments and for each, handshakes with its server through a series of function calls. During this phase, the SCM informs each playback server of the desired segment to stream back by providing its start time, duration, channel ID using the parameter data that was passed in earlier. Once again, if any part of the procedure fails, the entire initialization (and thus playback) is aborted. At the completion of this phase, every server should have loaded all the audio files associated with their portion of the entire data stream. Each is now ready for audio buffer requests.

The SCM then waits for a client to execute a "Start-Stream" call. In a graphical interface, this would occur, for example, when a user hits a Play button or begins a Save operation. Once this function is called, a separate thread spawns which will handle the entire process.

First, the current play position is checked to see which segment to begin playing on (a Move operation, explained below, controls the manual repositioning of this value). This is determined by looping through all of the segments, adding each segment's duration to a running total. When the current segment's duration added to the total exceeds the play position, that is the segment which contains the current play position.

Once this calculation is complete, a loop begins which starts from the previously determined segment and proceeds through the rest of the segment vector. For each segment, requests are formed for a predetermined buffer size and sent to the associated server. Once a buffer is returned, based on a flag configurable from the client, the SCM will either directly send back this data or "slice" it for the client first before returning it. Here, slicing refers to a process of dividing the buffer into smaller buffers by a least common

58

multiple known as a block align; this is sometimes useful to a client with a graphical component because the interface may need to reflect the amount played in smaller subdivisions.

When it is detected that all data from a segment has been requested, the SCM automatically steps to the next segment (possibly located on a different Voice Server) and begins requesting data from it instead. Because all Voice Servers are pre-loaded with the data and "ready to go," this process takes place in a fraction of a second, and the client does not sense any gap in the audio data being returned. In fact, the only true method for discerning the segment boundaries involves listening for normal, audible indicators of a transfer being made (clicking, ringing, or hearing the voice of a new participant) as provided through the telephone switch environment.

At the close of a play session (e.g., the user hits Stop or Pause in a typical audio playback GUI displayed in conjunction with the GUI described in FIG. 16) a StopStream call is made to the SCM. The thread in turn detects that the stopped state has been entered, exits from the request loop code, and frees up any used resources. Finally, it informs the client that a Stop event has occurred. If the entire call record is played without calling StopStream, the SCM performs the same exit and cleanup code, but informs the client that a Done event has occurred instead.

Movement within the overall stream is straightforward, given the aforementioned method that the SCM uses to determine which segment to begin playing from. A global variable holds the total number of milliseconds of audio data requested thus far. When a Move is performed, the server containing the data at the destination position is told to re-position itself, and the current play position is reset. Now, once StartStream executes again, it will initially start requesting from the server that was just moved to. And because that server had also moved its position pointer ahead, data will not be streamed from the beginning of the segment, but from where the Move position fell within that segment. Thus movement is a synchronized action completely transparent to the client, who is, ultimately, only interested in treating the data as a single stream.

SCM Pseudo-code

1. Initialize receives segment description data (start time, duration, etc.)
 - a.) Form a vector of all segments.
 - b.) Try to connect to all segments' servers.
 - c.) If there is an error connecting to any server, exit.
 - d.) Try to initialize each connected server.
 - e.) If there is an error initializing any server, exit.
2. If StartStream received:
 - a.) Go through segment list. Find segment of current play position.
 - b.) Starting with that segment, contact the associated server and begin requesting buffers.
 - c.) If option set, divide up buffer into smaller chunks.
 - d.) Send buffer(s) to client via event callback.
 - e.) Repeat until all data requested for this segment on that server.
 - f.) Repeat from step b. with next segment in list.
3. If Stop received:
 - a.) Exit from request loop.
 - b.) Clean up used resources.
 - c.) Send "Stop" event back to client.
4. If Stop not received, but all data from all segments played:
 - a.) Exit from request loop.

US 6,249,570 B1

59

- b.) Clean up used resources.
 - c.) Send "Done" event back to client.
5. Move received:
- a.) Go through segment list. Find segment of desired play position.
 - b.) Contact the associated server and reposition to that desired position.
 - c.) Reset current play position variable to reflect change.
- Detailed flow diagrams describing SCM operation are provided in FIGS. 20, 20A, 20B, 21, 22, 22A, 22B, and 22C. FIG. 20 illustrates the initialization process of the Stream Control Manager. The Initialization Sequence begins when a user enters the User Workstation playback software and at step 2010 queries for a recorded call record by desired criteria. At step 2012 a call record browser displays resulting call records. At step 2014 the user selects the desired record for playback. At step 2016 the browser invokes a PbkControlWin object: a dialog containing the 'player' ActiveX control.

At step 2020 the browser sends information to PbkControlWin about all segments comprising the call record. If at step 2024 immediate playback is not required, at step 2028 the entry is added to a playlist for future playback, and at step 2030 SUCCESS is returned. If at step 2024 immediate playback is required, at step 2032 the call record ID and segment list are forwarded to a GUI Player module. At step 2038 (see FIG. 20A) the player module instantiates a local SCM (StreamControl) object and stores a pointer in m_piStreamControl. At step 2040 the player module accepts the data, displays starting time and total duration (by parsing out string data), and forwards it to the final module, the Stream Control Manager (SCM), for audio playback.

Step 2046 begins the creation of a segments vector. At step 2046, a segment is parsed out from segList. At step 2048, recorder ID, start time, duration, and channel are parsed out from the segment. At step 2050, a new SEGMENT structure is created from recorder ID, start time, duration, and channel. At step 2052, a new SEGMENT is added to the SEGMENT vector. At step 2054, if all segments have been parsed from segList, at step 2058 an element is gotten from the SEGMENT vector. If at step 2054 more segments remain to be parsed from segList, steps 2046, 2048, 2050, and 2052 are repeated.

After step 2058, the program determines at step 2060 whether a new DCOM connection is required to the recorder for this segment. If not, at step 2062 the existing pointer is copied from the Connections vector to the server pointer in the SEGMENT vector and the program proceeds to step 2076. If at step 2060 the connection is new, a connection is made to the indicated recorder's "PlayBackServer" DCOM object using CoCreateInstanceEx. At step 2066 the program checks whether the object instantiated successfully. If not, at step 2068 a log error message occurs and at step 2070 ERROR (C) is returned. If at step 2066 the object instantiated successfully, at step 2072 (see FIG. 20B) the new object's pointer is added to the Connections vector. At step 2074 the program determines whether all segments have been connected. If not, the program returns to step 2058. If at step 2074 all segments have been connected, at step 2076 an element is gotten from the SEGMENT vector. At step 2078 the program queries for a list of wave files on the server that go with this segment. At step 2080 the program determines whether the query was successful. If not, at step 2082 a log error message occurs, and at step 2084 ERROR (C) is returned.

If at step 2080 the query was successful, at step 2088 the program opens the wave files on the server and prepares

60

them for streaming. It also returns the wave format of the audio in the segment. At step 2093 the program determines whether the wave files and format were obtained successfully. If not, at step 2094 a log error message occurs and at step 2095 ERROR (C) is returned. If step 2088 is determined at step 2093 to have been successful, at step 2096 the program checks whether all segments have been initialized. If not, the program returns to step 2076. If so, step 2097 is performed and at step 2098 SUCCESS is returned.

FIG. 21 illustrates how the program manages a Player Object 2110 and a PbkControlWin Object 2132.

FIG. 22 illustrates the playback sequence of the Stream Control Manager. Initially, at step 2202 a user has completed initialization and is waiting to hit Play in the Player GUI. At step 2204 the user hits the Play button. At step 2206 a message is sent to the Play method in the Player ActiveX control. At step 2210 the Play method in Player ActiveX control causes the output buffers to be "sliced" to increase the number of smaller buffers sent, thus increasing the resolution of the "totalPlayed" variable. At step 2218 Play method causes the server-side position to move to the current slider position. At step 2222 the program gets segment i++ from the SEGMENT vector. At step 2224 (see FIG. 22A) the program determines whether the End Time offset for segment i is greater than curPosition. If not, the program returns to step 2222. If so, the program proceeds to step 2226 and causes the file pointer on the server side to change to the appropriate new location. The program checks at step 2230 whether step 2226 was successful. If not, at step 2232 a log error message occurs and at step 2234 ERROR (C) is returned.

If at step 2230 step 2226 is determined to have been successful, at step 2238 the program calls StreamControl::StartStream. At step 2242 the program gets segment i++ from the SEGMENT vector. At step 2244 the program calls CoMarshalInterThreadInterfaceInStream to marshal a DCOM pointer member across a thread boundary. At step 2246 the program determines whether all SEGMENT elements have been marshaled. If not, the program returns to step 2242. If so, at step 2248 the main SCM streaming thread is spawned.

FIG. 22B illustrates an SCM main streaming thread. When the thread begins, at step 2250 the thread gets a segment from the SEGMENT vector. At step 2252 CoCellInterfaceAndReleaseStream is called to unmarshal a DCOM pointer member across the thread boundary. At step 2254 the thread checks whether all SEGMENT elements have been unmarshaled. If not, the thread returns to step 2250. If at step 2250 all SEGMENT elements are determined to have been unmarshaled, at step 2256 the thread gets a segment from the SEGMENT vector. The thread then checks at step 2258 whether the End Time offset for segment i is greater than curAmountRequested. If not, the thread returns to step 2256. If so, at step 2260 the thread gets Segment[i++]. The thread checks at step 2262 whether i is less than the highest segment number. If not, an Event::Done method is called at step 2264, and at step 2266 SUCCESS (C) is returned. If so, at step 2268 the thread determines whether this is the first segment to be played in this instance of the thread. If not, at step 2270 the thread calls PServer::PositionPlay(totalRequested) for Segment[i] and goes to step 2272. If so, the thread goes directly to step 2272.

At step 2272, the thread checks whether totalRequested is less than Segment[i].endTimeOffset. If not, the thread returns to step 2260. If so, the thread proceeds to step 2274 and checks whether totalRequested plus bufferSize is less

US 6,249,570 B1

61

than or equal to Segment[i].endTimeOffset. If not, at step 2276 the thread calculates a new bufferSize in multiples of the audio format's "block align." and proceeds to step 2278 (see FIG. 22C). If so, the thread proceeds directly to step 2278. At step 2278, the thread calls PBServer::ReqBuffer for Segment[i]. This is the core routine that actually retrieves a buffer of data from the PlayBack Server. At step 2286 the thread checks whether step 2278 was successful. If not, at step 2284 a log error message occurs, and at step 2282 ERROR (C) is returned.

If at step 2286 the thread determines that step 2278 was successful, at step 2287 totalRequested is set equal to totalRequested plus Actual returned buffer size. At step 2288, the thread checks whether BlockSlicing is enabled. If not, at step 2289 the thread sends the buffer back to the Player via Event::SendData method and returns to step 2274. If BlockSlicing has been enabled, at step 2292 the thread checks whether the CODEC is Dialogic OKI ADPCM or PCM. If not, at step 2293 the slice of the slices is set equal to the audio format's block align and the thread proceeds to step 2296. If so, at step 2294 the size of the slices is set to an even dividend of the buffer size (e.g., one-tenth of the buffer size). At step 2296, the thread copies out "slice size" from the buffer and sends it back to Player via Event::SendData method. At step 2298 the thread checks whether the entire buffer has been sent back. If not, the thread returns to step 2298. If so, the thread returns to step 2274.

The Stream Control Manager could theoretically be adapted to be used in more general streaming media situations, outside that of communications recording systems. In most current stream-based systems for network-based playback of audio content, such as RealMedia and NetShow, two general broadcast architectures exist known as unicast and multicast. Unicast involves a single client-server connection for data streaming, while in the multicast scenario a server pushes data to a single network address which multiple clients can then "tune in" to. However both models assume that data is being continuously fed from a single server. In the interest of load balancing, or if pieces of a streaming presentation were spread out across multiple locations, the SCM model could provide an innovative solution where the client side has the power to weave together many streams into a single playback session. An example could be imagined where a news organization, such as CNN, dynamically assembles a streaming broadcast for the online viewer from many different reports located on servers across the country. The components could be played seamlessly end-on-end using the SCM model, and if the viewer desired to rewind or fast-forward to a specific point in the stream, the SCM model would allow for complete transparent control.

The present invention is not to be limited in scope by the specific embodiments described herein. Indeed, modifications of the preferred embodiment in addition to those described herein will become apparent to those skilled in the art from the foregoing description and accompanying figures. Doubtless, numerous other embodiments can be conceived that would not depart from the teaching of the present invention, which scope is defined by the following claims.

All the features disclosed in this specification (including any accompanying claims, abstract, and drawings) may be replaced by alternative features serving the same, equivalent, or similar purpose, unless expressly stated otherwise. Thus, unless expressly stated otherwise, each feature disclosed is one example only of a generic series of equivalent or similar features.

62

What is claimed is:

1. A system for recording information regarding telephone calls comprising one or more segments, comprising:

- (a) a first memory having one or more locations storing audio data regarding telephone call segments relating to one or more telephone calls;
- (b) a second memory having one or more locations storing data regarding telephony events associated with the telephone call segments; and
- (c) a processor programmed to identify telephone call segments that relate to one telephone call and to construct a data representation of a lifetime of the telephone call, using data regarding telephony events associated with the telephone call segments of the telephone call, wherein said data representation comprises
 - (i) a list of participants in the telephone call,
 - (ii) a list of telephony events regarding the call,
 - (iii) a list containing the time each telephony event occurred,
 - (iv) the start and end time of the call, and
 - (v) the start time, end time, and duration of each telephone call segment.

2. A system for recording information regarding telephone calls comprising one or more segments, comprising:

- (a) a first memory having one or more locations storing audio data regarding telephone call segments relating to one or more telephone calls;
- (b) a second memory having one or more locations storing data regarding telephony events associated with the telephone call segments; and
- (c) a processor programmed to identify telephone call segments that relate to one telephone call and to construct a data representation of a lifetime of the telephone call, using data regarding telephony events associated with the telephone call segments of the telephone call, wherein the data representation comprises, for each segment of the call, the location of the stored audio data of the segment and the start time, end time, and duration of each telephone call segment.

3. A system for recording information regarding telephone calls comprising one or more segments, comprising:

- (a) a first memory having one or more locations storing audio data regarding telephone call segments relating to one or more telephone calls;
- (b) a second memory having one or more locations storing data regarding telephony events associated with the telephone call segments, said data regarding telephony events being received from a plurality of sources connected to a telephone switching environment, and wherein at least one of the sources is a real-time link and at least one of the sources is not a real-time link; and

- (c) a processor programmed to identify telephone call segments that relate to one telephone call and to construct a data representation of a lifetime of the telephone call, using data regarding telephony events associated with the telephone call segments of the telephone call.

4. The system of claim 3 wherein at least one of the sources connected to a telephone switching environment is a CTI link and at least one of the sources is an SMDR link.

5. A method for recording information regarding telephone calls comprising one or more segments, comprising:

- (a) receiving audio data regarding one or more telephone call segments relating to one or more telephone calls,

US 6,249,570 B1

63

- and data regarding telephony events associated with said telephone call segments;
- (b) storing the received audio data regarding telephone call segments;
- (c) storing the received data regarding telephony events associated with said telephone call segments;
- (d) identifying telephone call segments that relate to one telephone call; and
- (e) constructing a data representation of a lifetime of the telephone call using data regarding telephony events associated with the telephone call segments of the telephone call, wherein the data representation comprises
- (i) a list of participants in the telephone call,
 - (ii) a list of telephony events regarding the call,
 - (iii) a list containing the time each telephony event occurred,
 - (iv) the start and end time of the call, and
 - (v) the start time, end time, and duration of each telephone call segment.
6. A method for recording information regarding telephone calls comprising one or more segments, comprising:
- (a) receiving audio data regarding one or more telephone call segments relating to one or more telephone calls, and data regarding telephony events associated with said telephone call segments;
 - (b) storing the received audio data regarding telephone call segments;
 - (c) storing the received data regarding telephony events associated with said telephone call segments;
 - (d) identifying telephone call segments that relate to one telephone call; and
 - (e) constructing a data representation of a lifetime of the telephone call using data regarding telephony events

64

- associated with the telephone call segments of the telephone call, wherein said data representation comprises, for each segment of the call, the location of the stored audio data of that segment and the start time, end time, and duration of that segment.
7. A method for recording information regarding telephone calls comprising one or more segments, comprising:
- (a) receiving audio data regarding one or more telephone call segments relating to one or more telephone calls, and data regarding telephony events associated with said telephone call segments, wherein the data regarding telephony events is received from a plurality of sources connected to a telephone switching environment, wherein at least one of the sources is a real-time link and at least one of the sources is not a real-time link;
 - (b) storing the received audio data regarding telephone call segments;
 - (c) storing the received data regarding telephony events associated with said telephone call segments;
 - (d) identifying telephone call segments that relate to one telephone call; and
 - (e) constructing a data representation of a lifetime of the telephone call using data regarding telephony events associated with the telephone call segments of the telephone call.
8. The method of claim 7 wherein at least one of the sources connected to a telephone switching environment is a CTI link and at least one of the sources connected to a telephone switching environment is an SMDR link.

* * * * *

EXHIBIT E



US006728345B2

(12) **United States Patent**
Glowny et al.

(10) **Patent No.:** **US 6,728,345 B2**
(45) **Date of Patent:** **Apr. 27, 2004**

(54) **SYSTEM AND METHOD FOR RECORDING
AND STORING TELEPHONE CALL
INFORMATION**

(75) Inventors: **David A. Glowny**, Milford, CT (US);
Phil Min Ni, Danbury, CT (US); **John
E. Richter**, Trumbull, CT (US)

(73) Assignee: **Dictaphone Corporation**, Stratford, CT
(US)

(*) Notice: Subject to any disclaimer, the term of this
patent is extended or adjusted under 35
U.S.C. 154(b) by 385 days.

(21) Appl. No.: **09/876,979**

(22) Filed: **Jun. 8, 2001**

(65) **Prior Publication Data**

US 2001/0040942 A1 Nov. 15, 2001

Related U.S. Application Data

(63) Continuation of application No. 09/328,299, filed on Jun. 8,
1999, now Pat. No. 6,249,570.

(51) **Int. Cl.**⁷ **H04M 1/04**

(52) **U.S. Cl.** **379/88.22; 379/111; 379/202.01**

(58) **Field of Search** **379/67.1, 68, 88.09,**
379/88.11, 88.22, 88.25, 93.12, 93.17, 93.23,
111, 112.01, 112.06, 116, 202.01, 265.03,
265.07, 266.1, 267, 309; 360/5, 6, 55

(56) **References Cited**

U.S. PATENT DOCUMENTS

3,104,284 A * 9/1963 French et al. 704/203
3,369,077 A * 2/1968 French et al. 704/207
3,723,667 A * 3/1973 Park et al. 369/47.55
3,727,203 A * 4/1973 Crossman 360/72.2
4,130,739 A * 12/1978 Patten 369/47.36
4,199,820 A * 4/1980 Ohtake et al. 386/126
4,260,854 A * 4/1981 Kolodny et al. 379/75
4,298,954 A * 11/1981 Bigelow et al. 710/53

4,435,832 A * 3/1984 Asada et al. 704/262
4,442,485 A * 4/1984 Ota et al. 710/57
4,542,427 A * 9/1985 Nagai 360/72.1
4,549,047 A * 10/1985 Brian et al. 379/88.26
4,602,331 A * 7/1986 Sheth 711/113
4,621,357 A * 11/1986 Naiman et al. 370/370
4,630,261 A * 12/1986 Irvin 370/235
4,631,191 A * 12/1986 Dale et al. 424/186.1
4,679,191 A * 7/1987 Nelson et al. 370/355
4,686,587 A * 8/1987 Hipp et al. 360/74.2
4,688,117 A * 8/1987 Dwyer et al. 360/72.3
4,692,819 A * 9/1987 Steele 360/72.1
4,709,390 A * 11/1987 Atal et al. 704/262
4,785,408 A * 11/1988 Britton et al. 704/270
4,785,473 A * 11/1988 Pfeiffer et al. 379/88.24
4,799,144 A * 1/1989 Parruck et al. 710/2
4,799,217 A * 1/1989 Fang 370/458
4,811,131 A * 3/1989 Sander et al. 360/74.4
4,811,376 A * 3/1989 Davis et al. 340/7.28
4,827,461 A * 5/1989 Sander 369/7
4,829,514 A * 5/1989 Frimmel et al. 370/368
4,835,630 A * 5/1989 Freer 360/69
4,841,387 A * 6/1989 Rindfuss 360/72.1
4,851,937 A * 7/1989 Sander 360/69
4,853,952 A * 8/1989 Jachmann et al. 379/88.11
4,864,620 A * 9/1989 Bialick 704/207
4,873,589 A * 10/1989 Inazawa et al. 360/53

(List continued on next page.)

Primary Examiner—Scott L. Weaver

(74) *Attorney, Agent, or Firm*—Howrey Simon Arnold &
White, LLP; Anthony L. Meola

(57) **ABSTRACT**

A system and method for monitoring a telephone switching environment. In a preferred embodiment the system and method identify telephone call segments that relate to one telephone call and construct a data representation of a lifetime of the telephone call, using data regarding telephony events associated with the telephone call segments of the telephone call. The system and method are also capable of using the data representation to display a graphical representation of a lifetime of a telephone call.

52 Claims, 29 Drawing Sheets

AgentId	EXTENSION	LOCATION	START TIME	END TIME	CONNECT REASON	DISCONNECT REASON
A		EXTERNAL	t ₁	t ₇	NORM START	NORM DROP
B	0001	INTERNAL	t ₁	t ₄	NORM START	Xfr AWAY
HOLD		INTERNAL	t ₄	t ₆	Xfr Rec	Xfr AWAY
C	0002	INTERNAL	t ₆	t ₇	Xfr Rec	OTHER PARTY HANGUP

US 6,728,345 B2

Page 2

U.S. PATENT DOCUMENTS

4,890,325 A	*	12/1989	Taniguchi et al.	704/208	5,216,744 A	*	6/1993	Alleyne et al.	704/200
4,891,835 A	*	1/1990	Leung et al.	379/88.11	5,270,877 A	*	12/1993	Fukushima et al.	360/48
4,893,197 A	*	1/1990	Howells et al.	360/8	5,274,738 A	*	12/1993	Daly et al.	704/200
4,907,225 A	*	3/1990	Gulick et al.	370/463	5,283,818 A	*	2/1994	Klausner et al.	379/88.25
4,939,595 A	*	7/1990	Yoshimoto et al.	386/100	5,305,375 A	*	4/1994	Sagara et al.	379/88.27
4,975,941 A	*	12/1990	Morganstein et al.	379/88.23	5,339,203 A	*	8/1994	Henits et al.	360/39
4,991,217 A	*	2/1991	Garrett et al.	704/235	5,353,168 A	*	10/1994	Crick	360/5
5,001,703 A	*	3/1991	Johnson et al.	370/280	5,396,371 A	*	3/1995	Henits et al.	360/5
5,031,146 A	*	7/1991	Umina et al.	365/189.01	5,404,455 A	*	4/1995	Daly et al.	710/4
5,065,428 A	*	11/1991	Mitchell et al.	713/194	5,446,603 A	*	8/1995	Henits et al.	360/48
5,129,036 A	*	7/1992	Dean et al.	704/200	5,457,782 A	*	10/1995	Daly et al.	704/200
5,130,975 A	*	7/1992	Akata	370/416	5,533,103 A	*	7/1996	Peavey et al.	379/69
5,142,527 A	*	8/1992	Barbier et al.	370/270	5,819,005 A	*	10/1998	Daly et al.	704/200
5,163,132 A	*	11/1992	DuLac et al.	710/53	5,867,559 A	*	2/1999	Jorgensen et al.	379/67.1
5,179,479 A	*	1/1993	Ahn	360/72.1	5,982,857 A	*	11/1999	Brady	379/88.19
5,195,128 A	*	3/1993	Knitl	379/88.24	6,070,241 A	*	5/2000	Edwards et al.	713/200
5,210,851 A	*	5/1993	Kato et al.	360/48					

* cited by examiner

U.S. Patent

Apr. 27, 2004

Sheet 1 of 29

US 6,728,345 B2

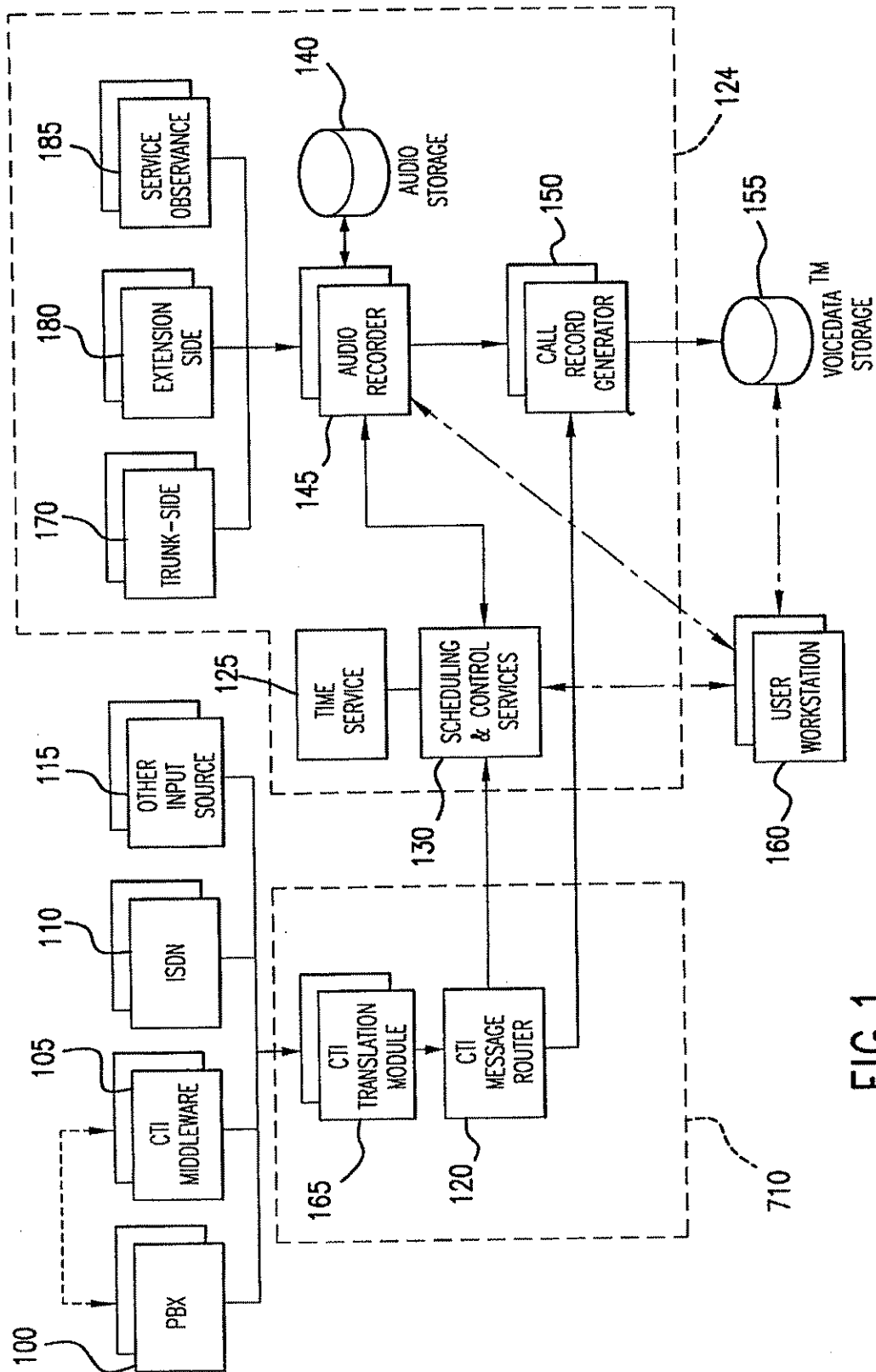


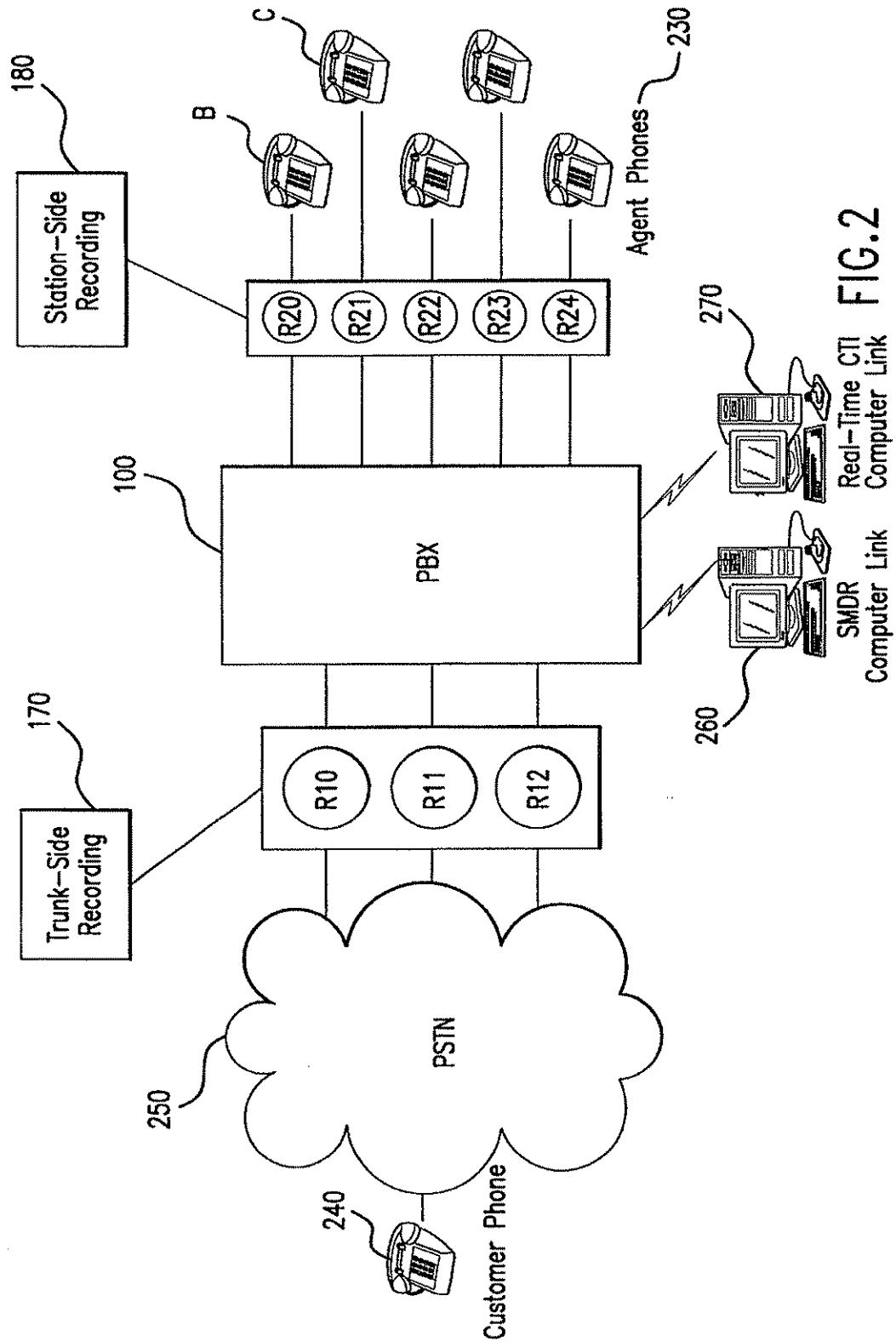
FIG. 1

U.S. Patent

Apr. 27, 2004

Sheet 2 of 29

US 6,728,345 B2



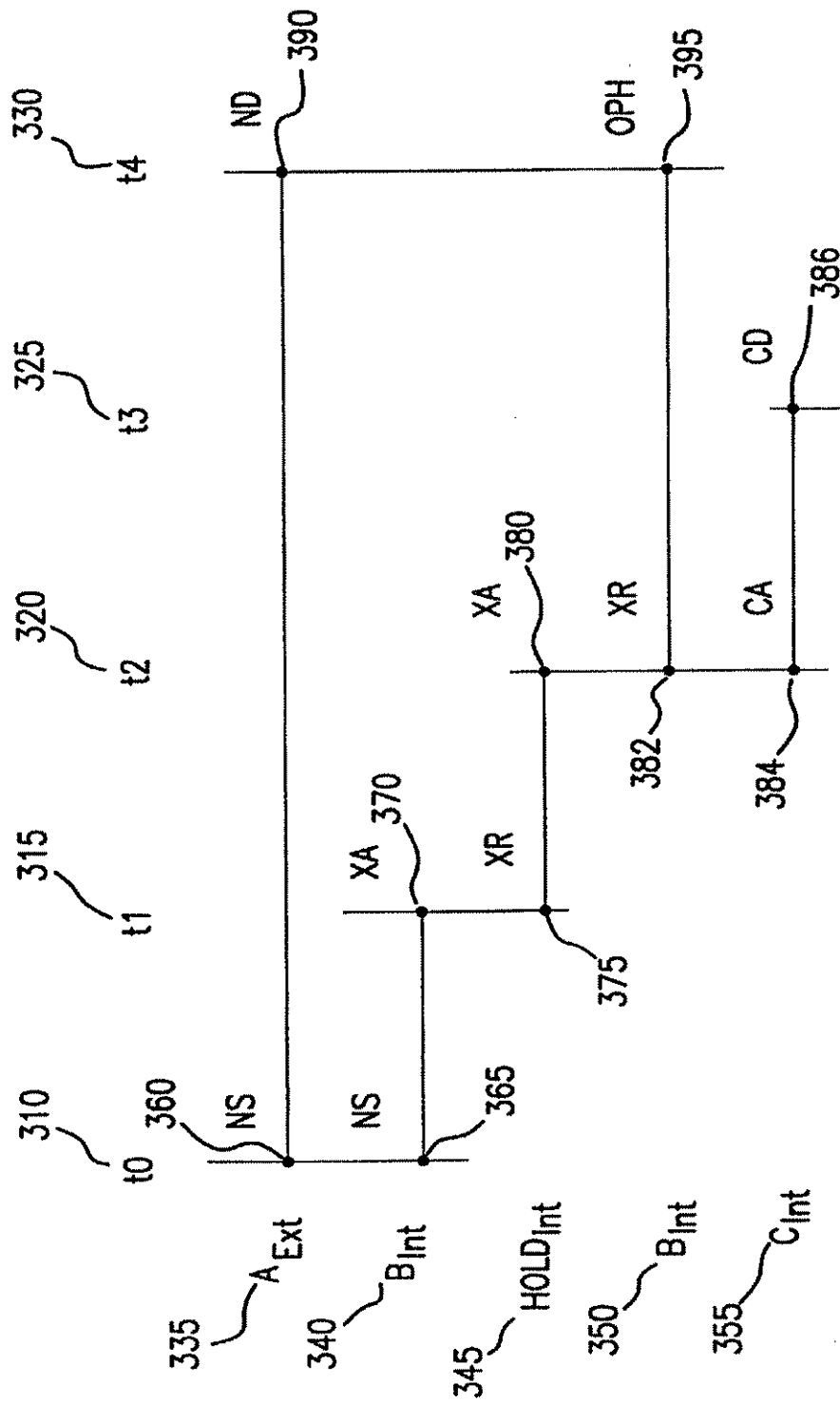


FIG. 3

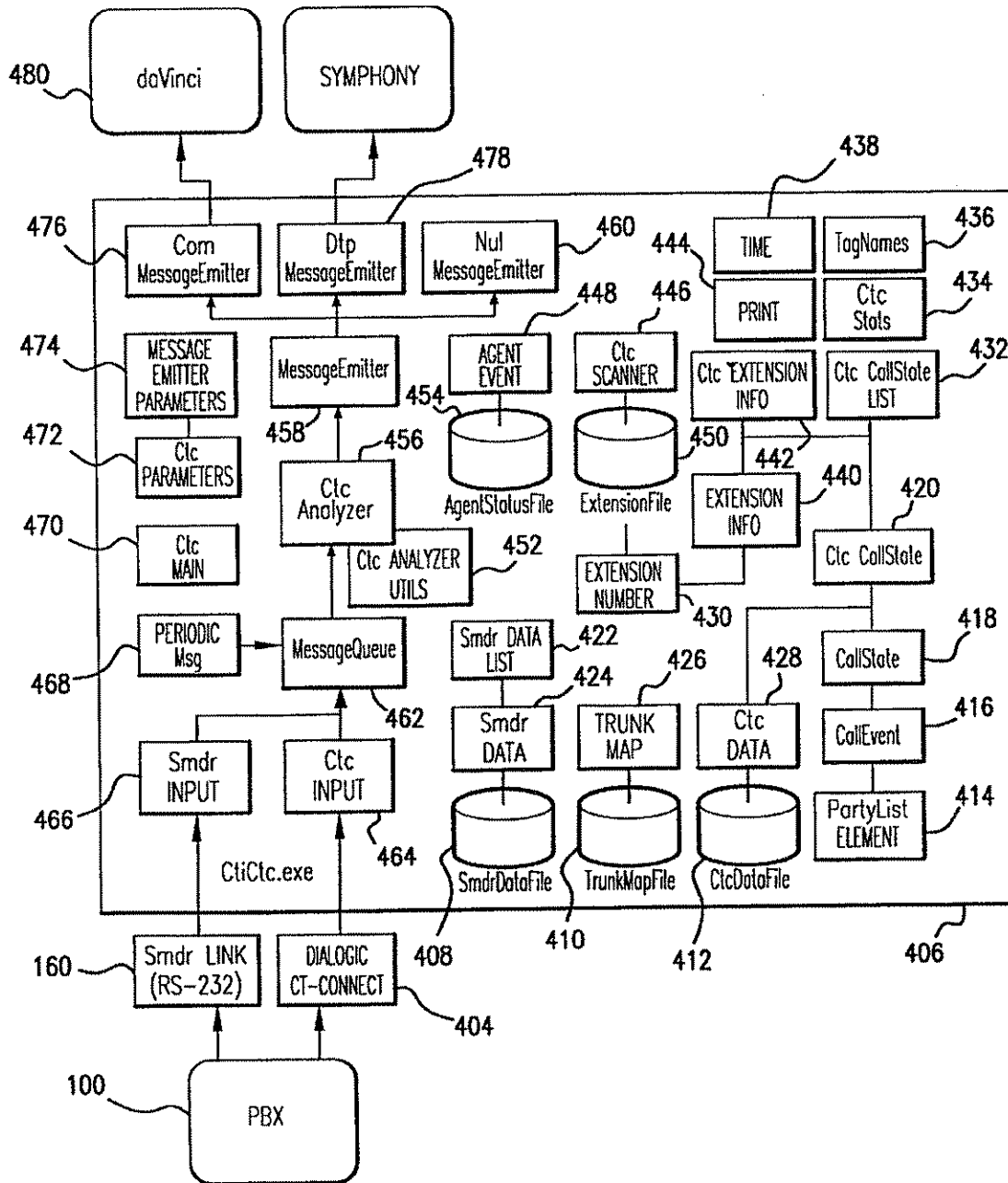


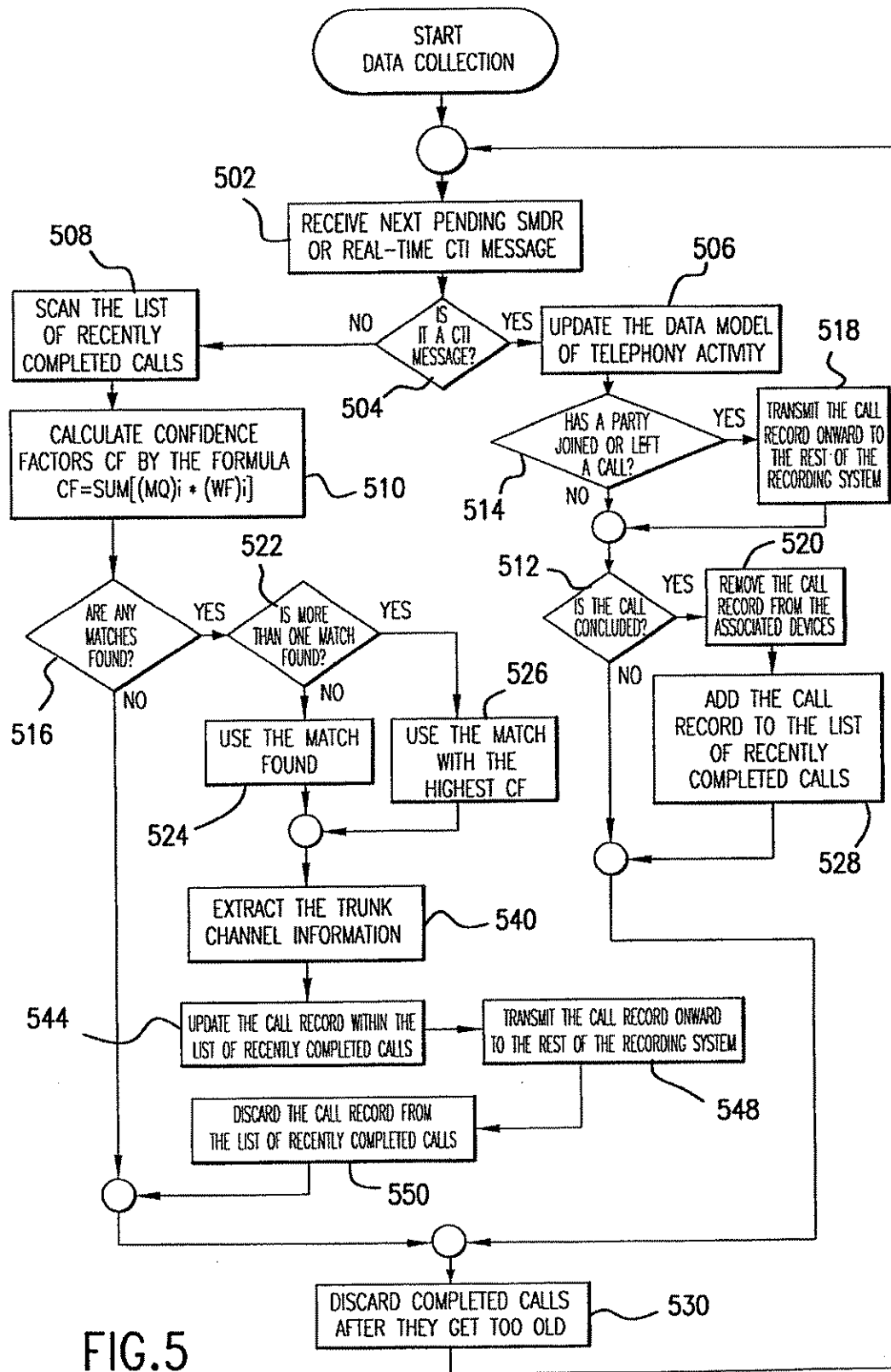
FIG.4

U.S. Patent

Apr. 27, 2004

Sheet 5 of 29

US 6,728,345 B2



U.S. Patent

Apr. 27, 2004

Sheet 6 of 29

US 6,728,345 B2

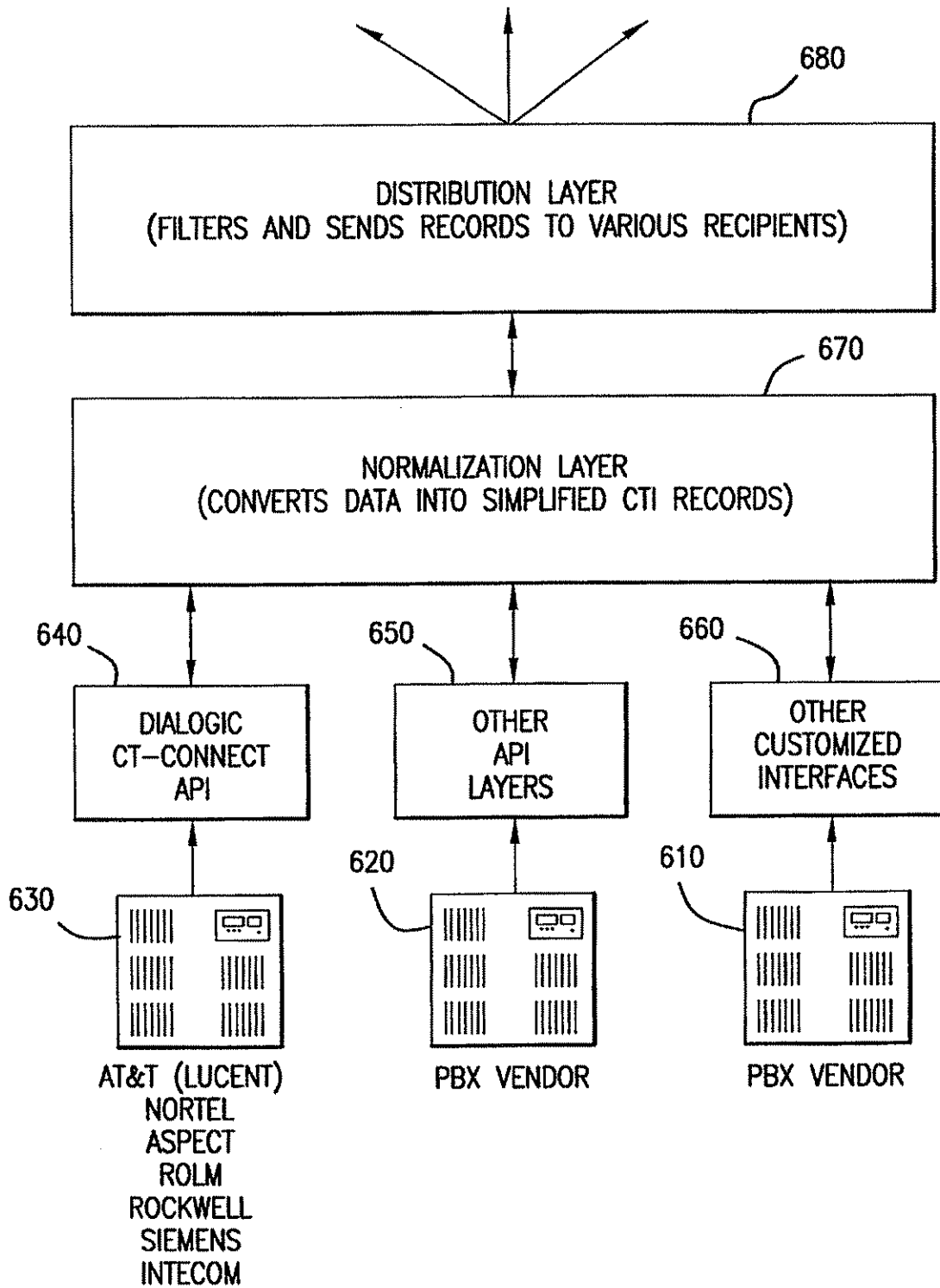


FIG.6

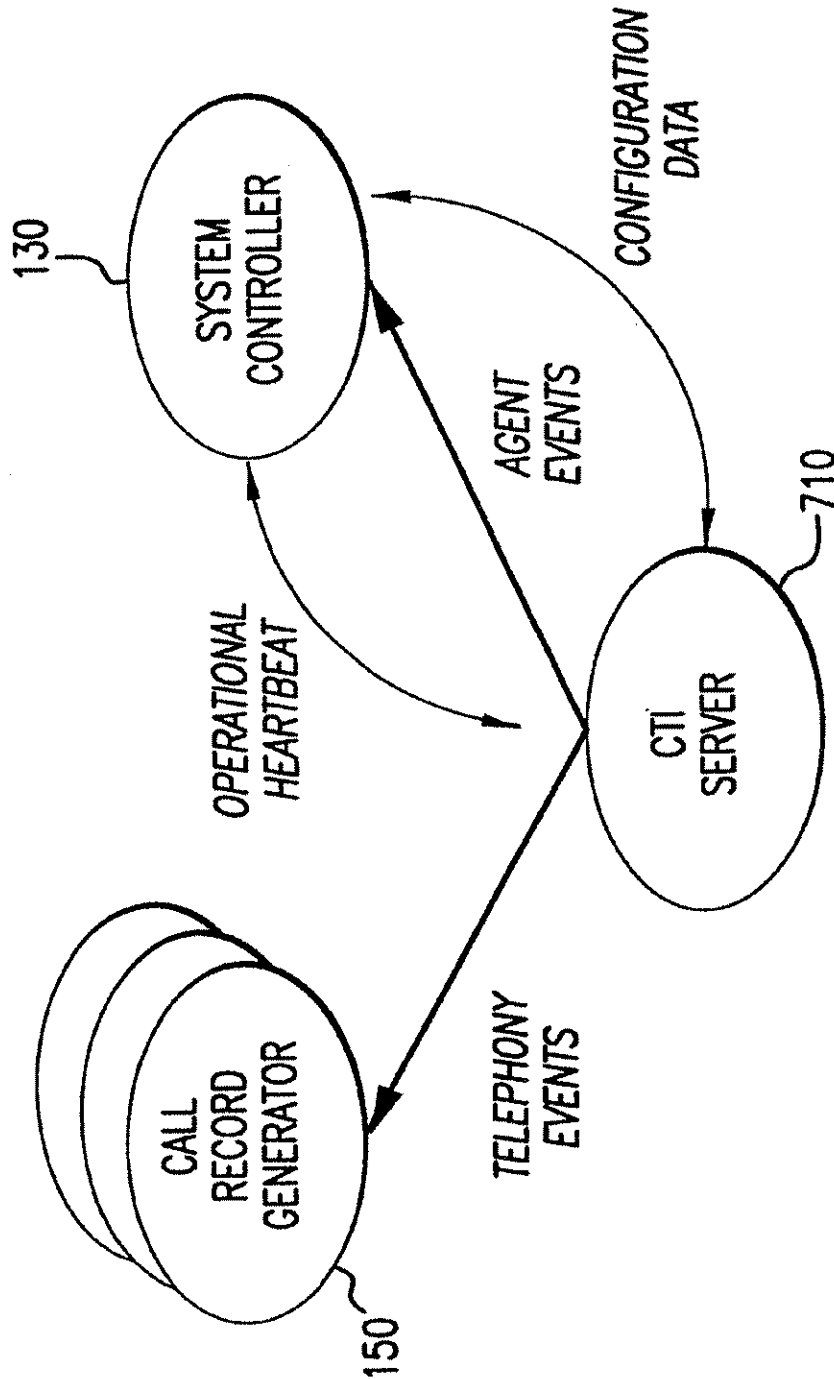


FIG. 7

U.S. Patent

Apr. 27, 2004

Sheet 8 of 29

US 6,728,345 B2

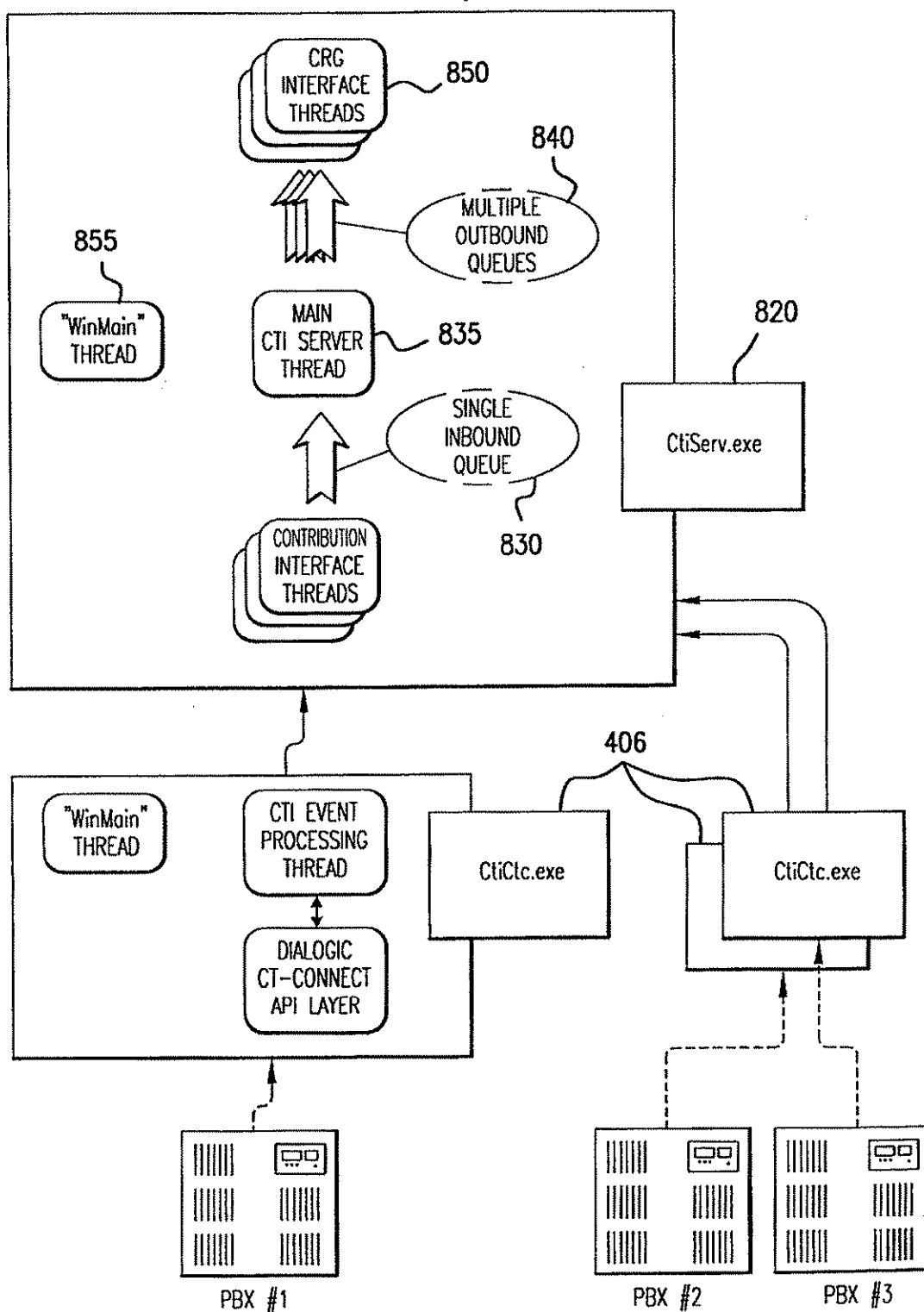


FIG. 8

U.S. Patent

Apr. 27, 2004

Sheet 9 of 29

US 6,728,345 B2

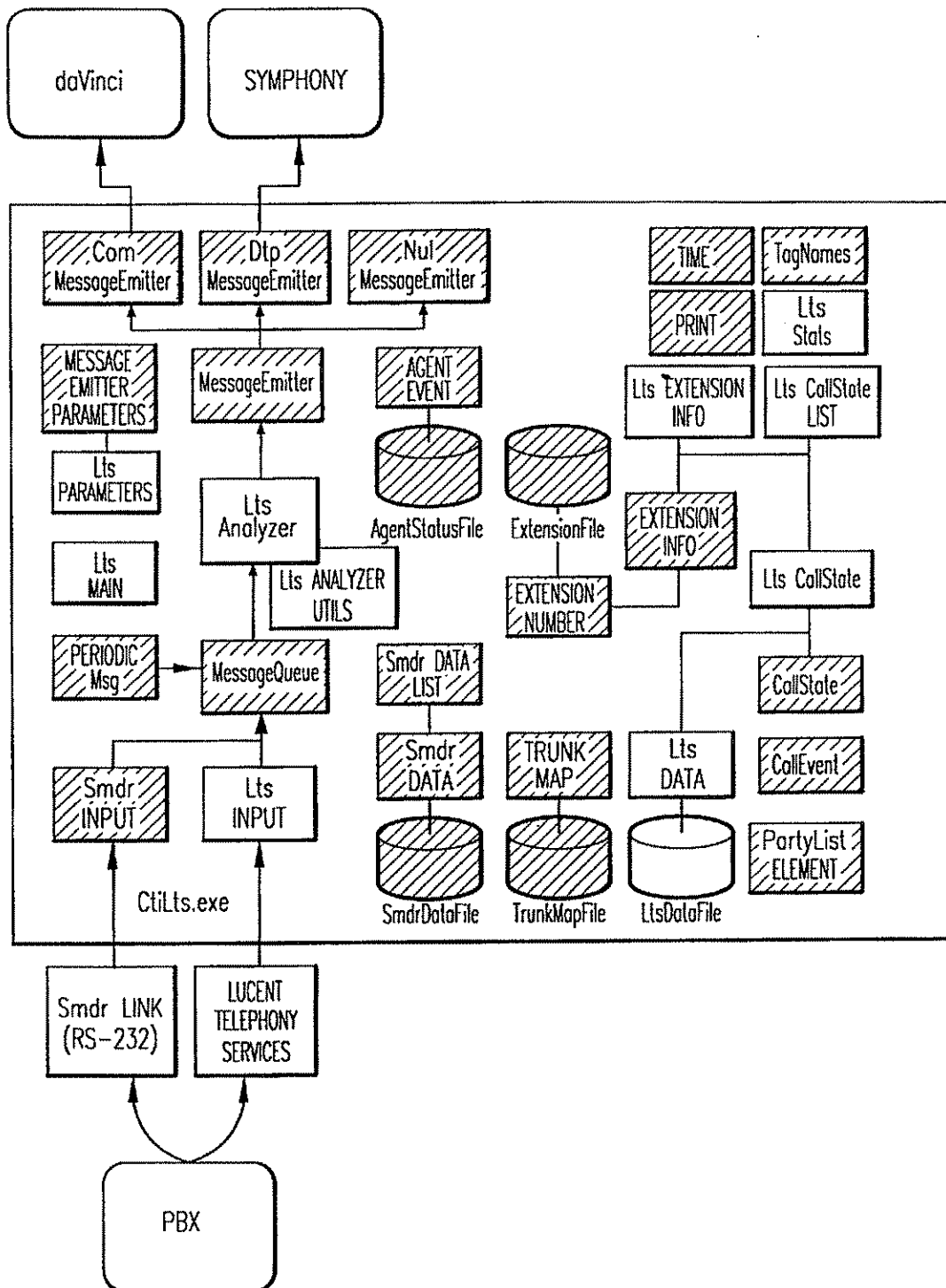
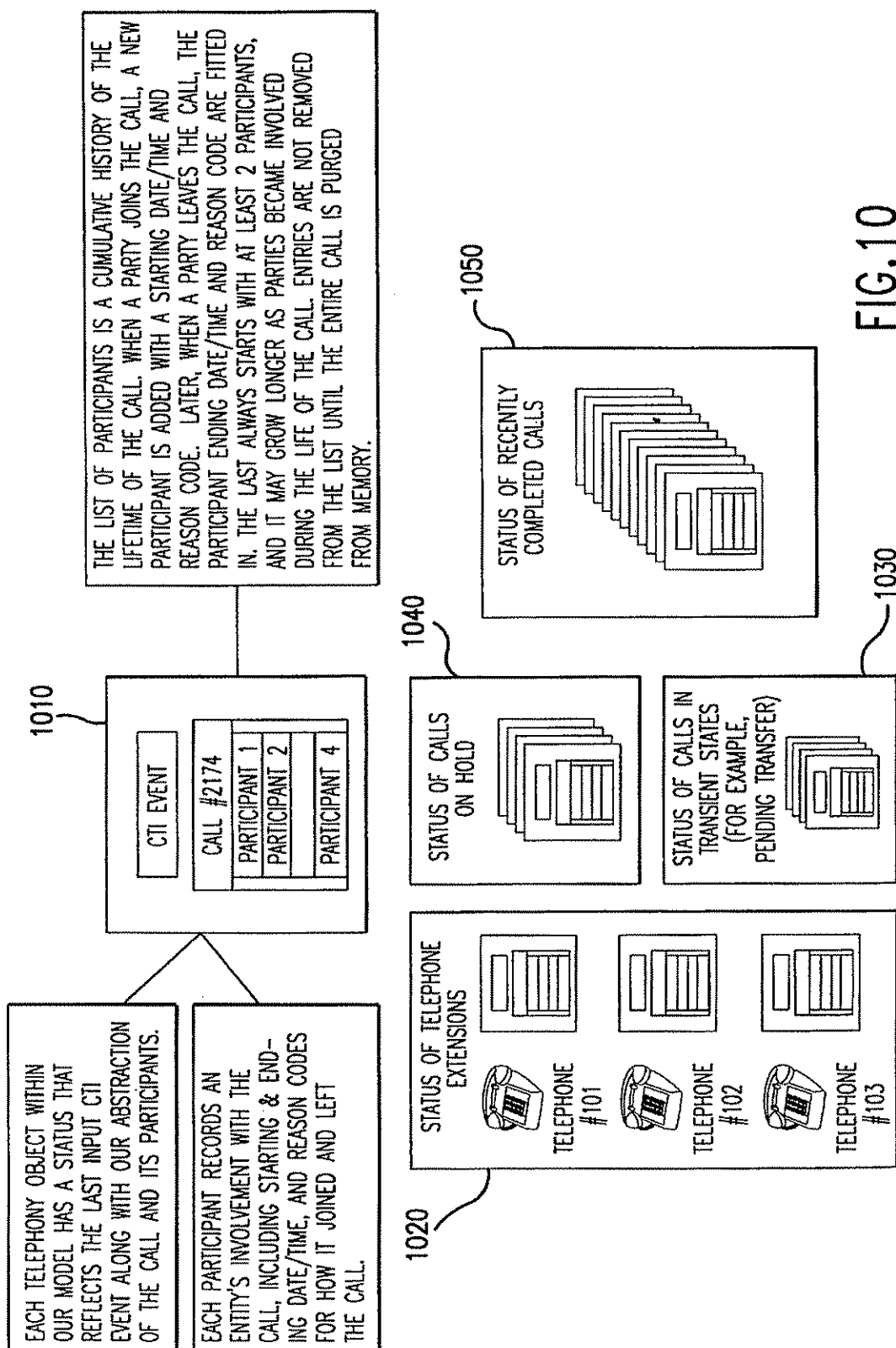


FIG.9



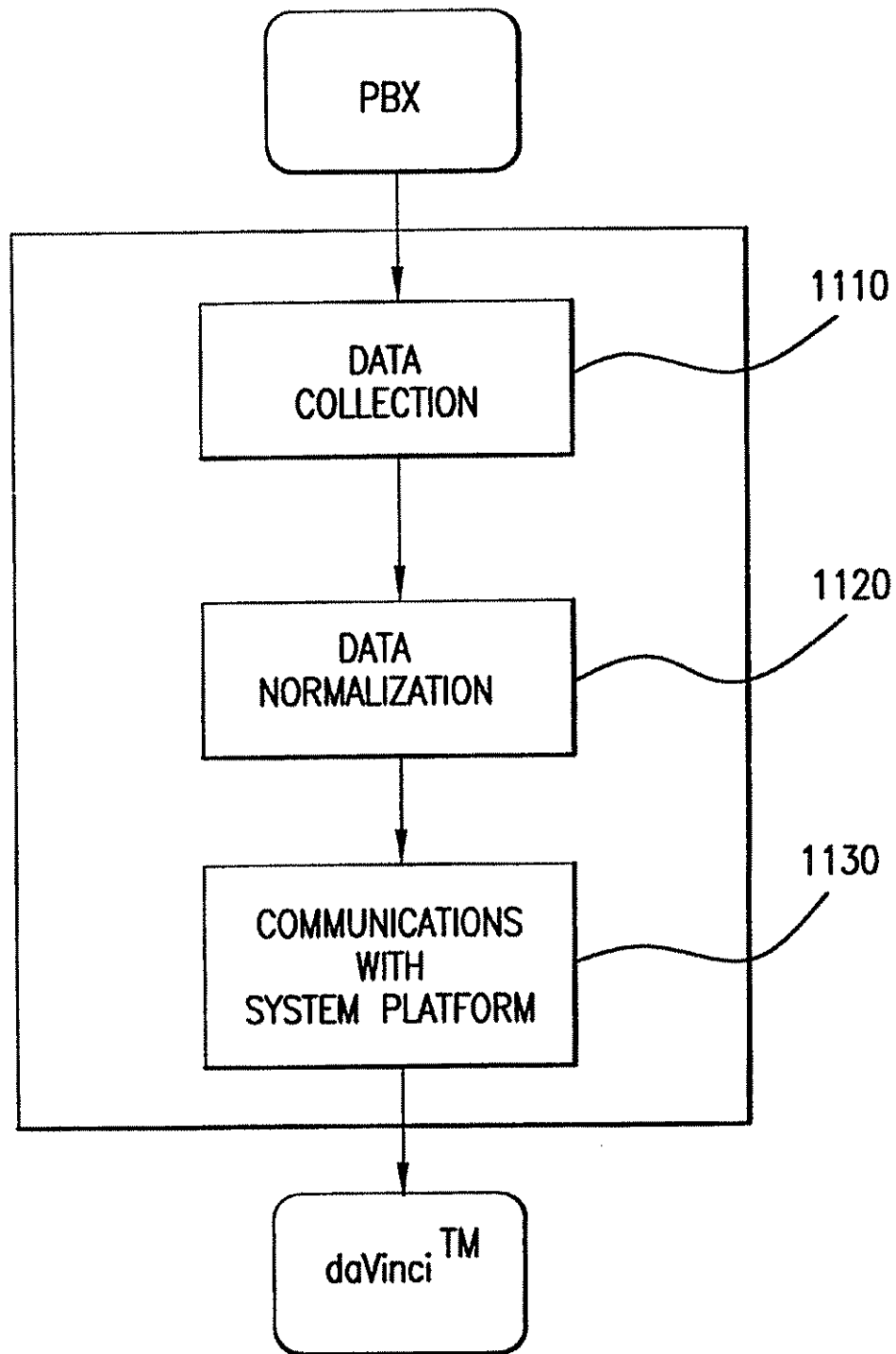


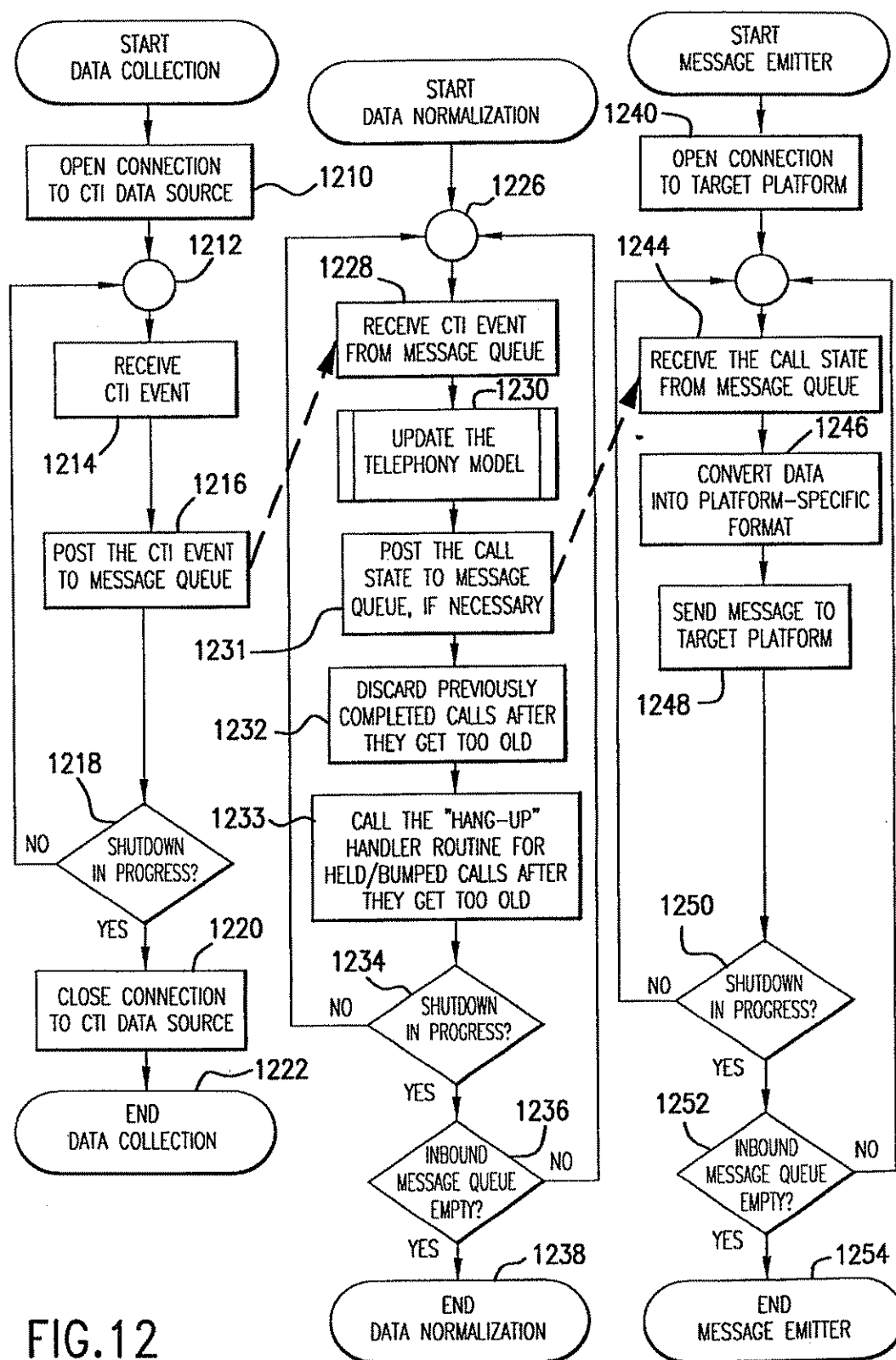
FIG.11

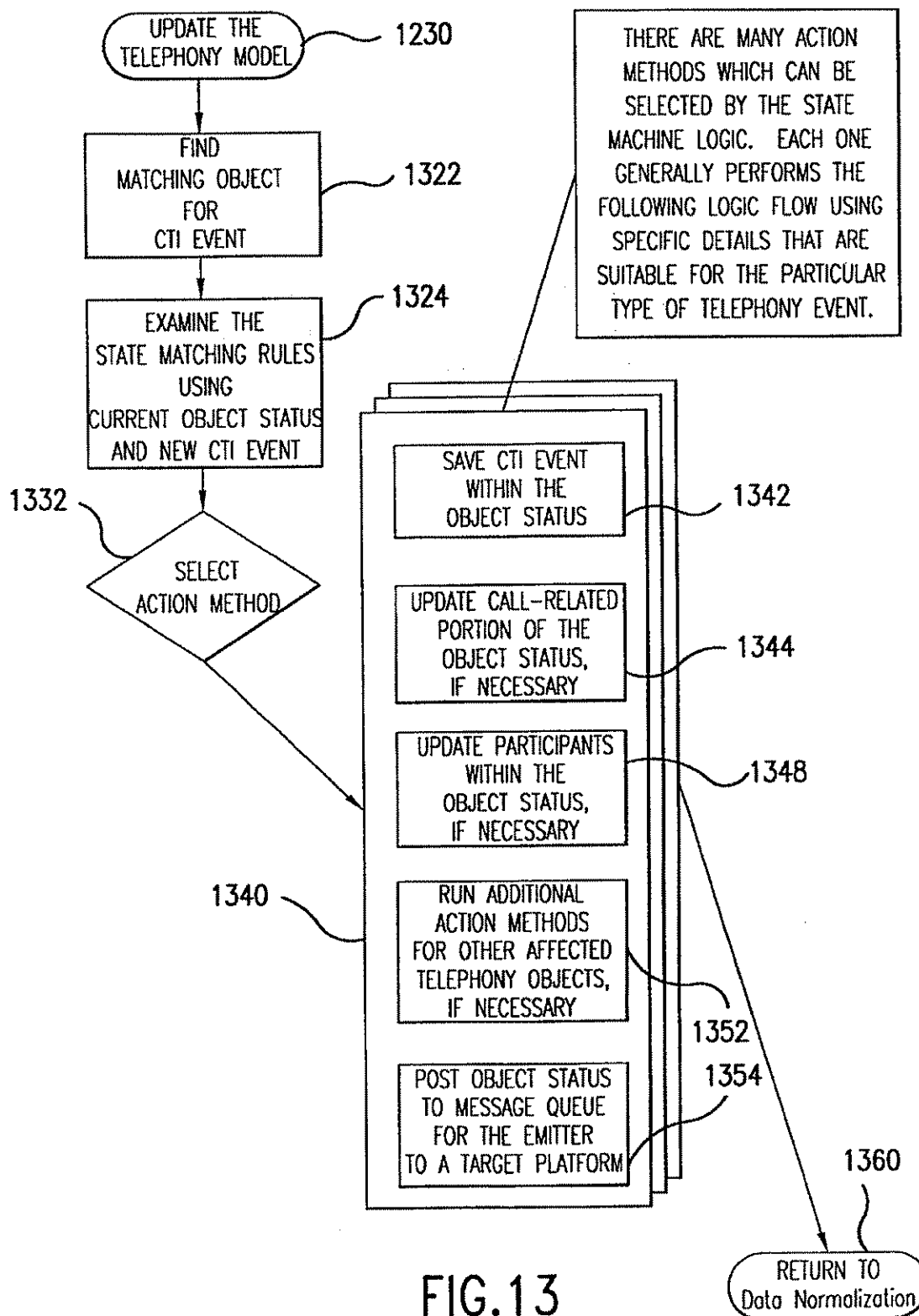
U.S. Patent

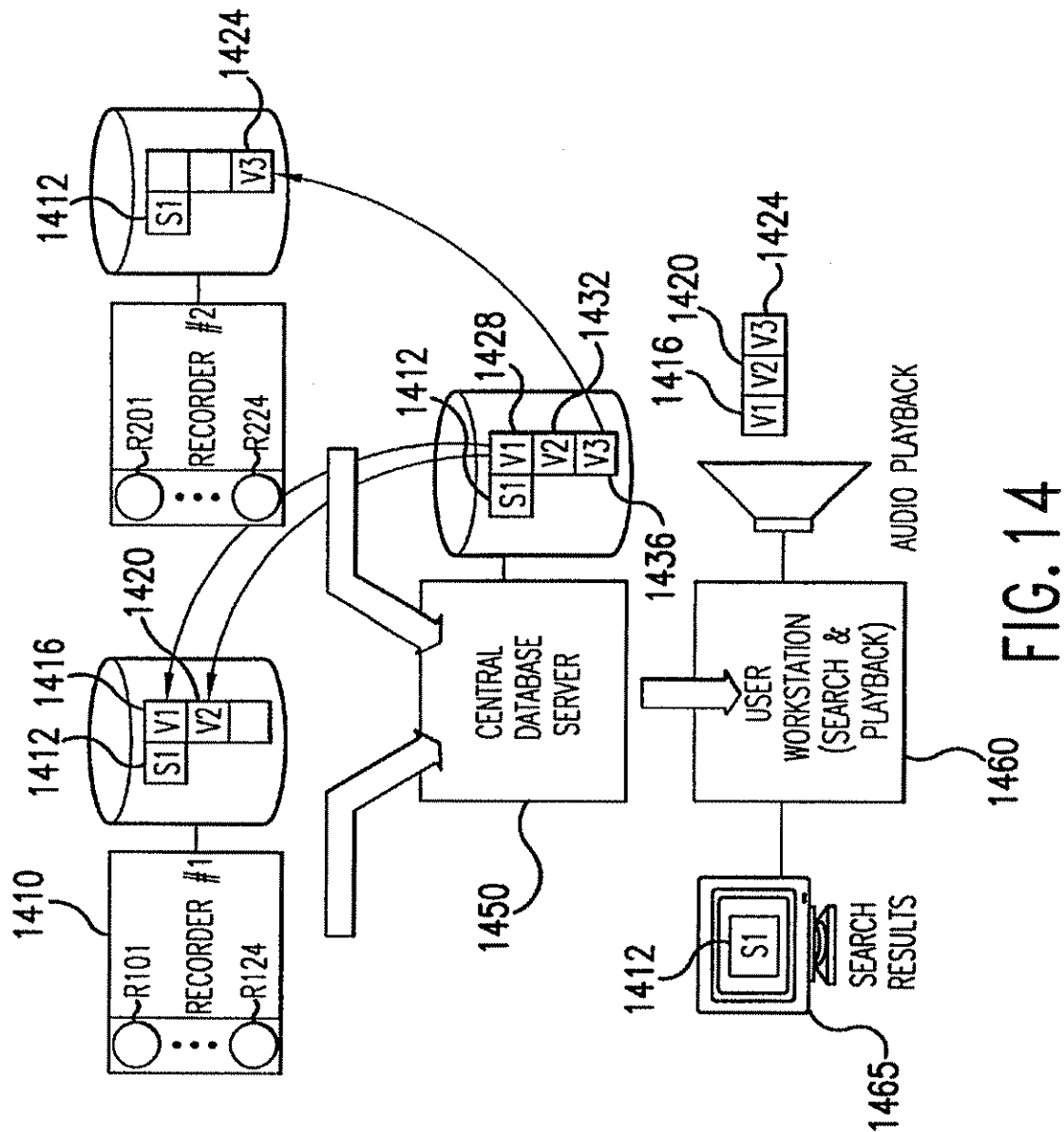
Apr. 27, 2004

Sheet 12 of 29

US 6,728,345 B2







U.S. Patent

Apr. 27, 2004

Sheet 15 of 29

US 6,728,345 B2

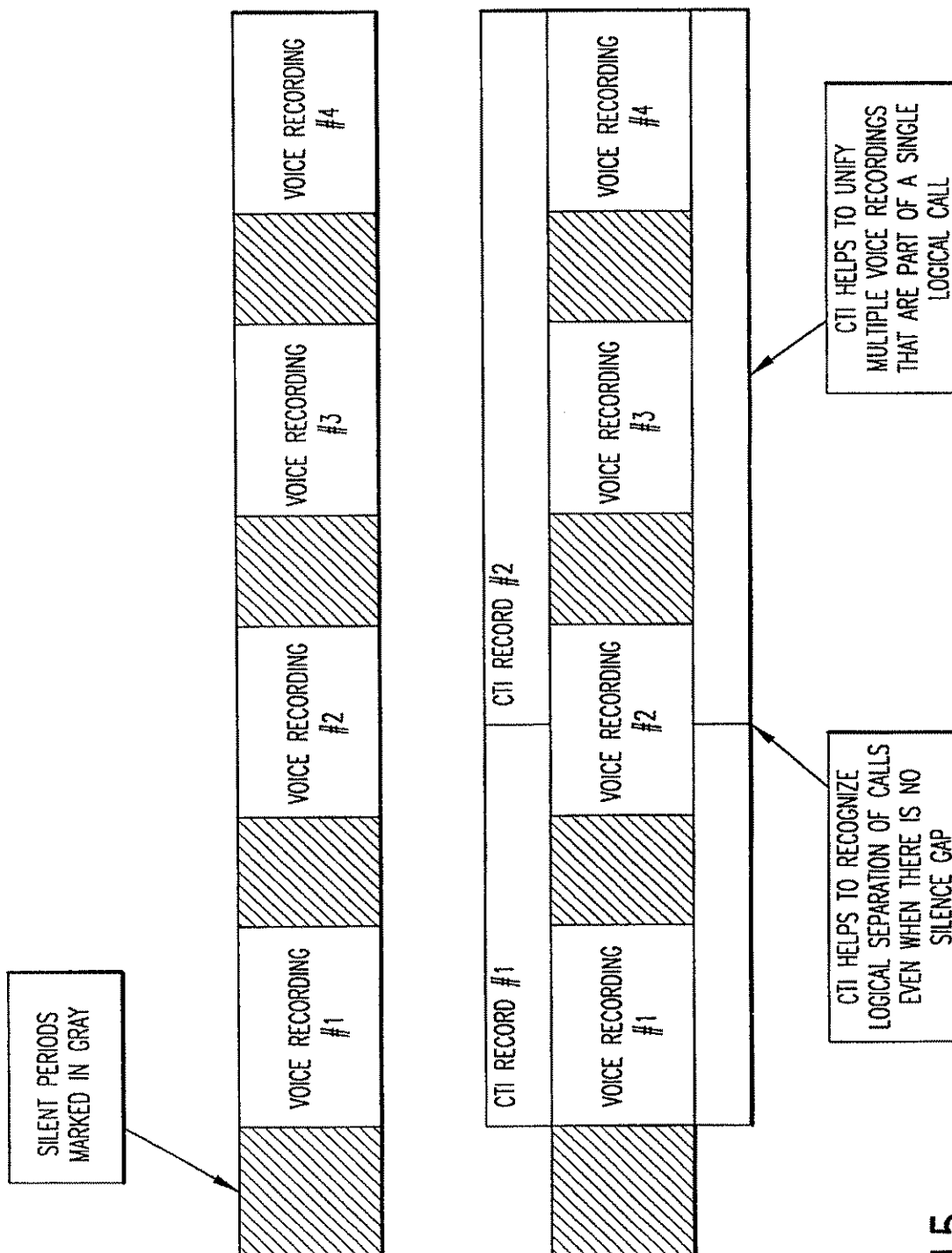


FIG.15

V da Vinci - [CALL RECORDS] [PHONE CALL: 765]

☐ FILE ☐ EDIT ☐ VIEW ☐ CALL RECORD ☐ TOOLS ☐ WINDOW ☐ HELP

ALL AGENTS	CALL DIRECTION	CALL DURATION	CALL FEATURE	CALL LOCKED	CALL ID	CALL START TIME	CALL STATUS	CALL TYPE	DIMF CODES
<input type="radio"/> AHMAD NASS	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> ALAN MACDO	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BARBARA BIS	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BARNEY FILE	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BETTY CRABLE	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BILL NOCANDO	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BILL STEWART	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BILLY MURRY	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BOB OSBORNE	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> BUFFY VAMP	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> CHRISTINE RAT	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H
<input type="radio"/> CINDY KALJAZ	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	BACKUP	NO MEDIA	H
<input type="radio"/> DAVE GORD	OUTBOUND	00:02:02	TRANSFER ONLY CALL OFF		147663	5/11/99 6:21:08 AM	NORMAL	MEDIA	H

☐ CALL RECORDS... 1620

START TIME	END TIME	DURATION	AGENT	CONNECTION TYPE	DISCONNECTION TYPE
5/11/99 6:20:08 AM	5/11/99 9:25:31 AM	03:05:23			
5/11/99 6:20:08 AM	5/11/99 6:55:31 AM	00:35:25	104	NORMAL START	TRANSFER AWAY
5/11/99 6:20:08 AM	5/11/99 9:08:51 AM	02:48:43	112	NORMAL START	CONFERENCE DROP
5/11/99 6:55:31 AM	5/11/99 7:10:22 AM	00:14:51	HOLD	TRANSFER RECEIVE	TRANSFER AWAY
5/11/99 7:10:22 AM	5/11/99 9:25:07 AM	02:14:45	112	TRANSFER RECEIVE	NORMAL DROP
5/11/99 7:10:22 AM	5/11/99 9:25:07 AM	02:14:45	134	CONFERENCE ADD	OTHER PARTY HANGUP

1610

FOR HELP, PRESS F1

☐ START ☐ DeVinciWorks... ☐ CONTROL PANEL ☐ MICROSOFT ACC... ☐ EXPLORING-W... ☐ C:\Dev\deVin... ☒ de Vinci-L... ☐ CALCULATOR ☐ 3:01 PM

FIG. 16

U.S. Patent

Apr. 27, 2004

Sheet 17 of 29

US 6,728,345 B2

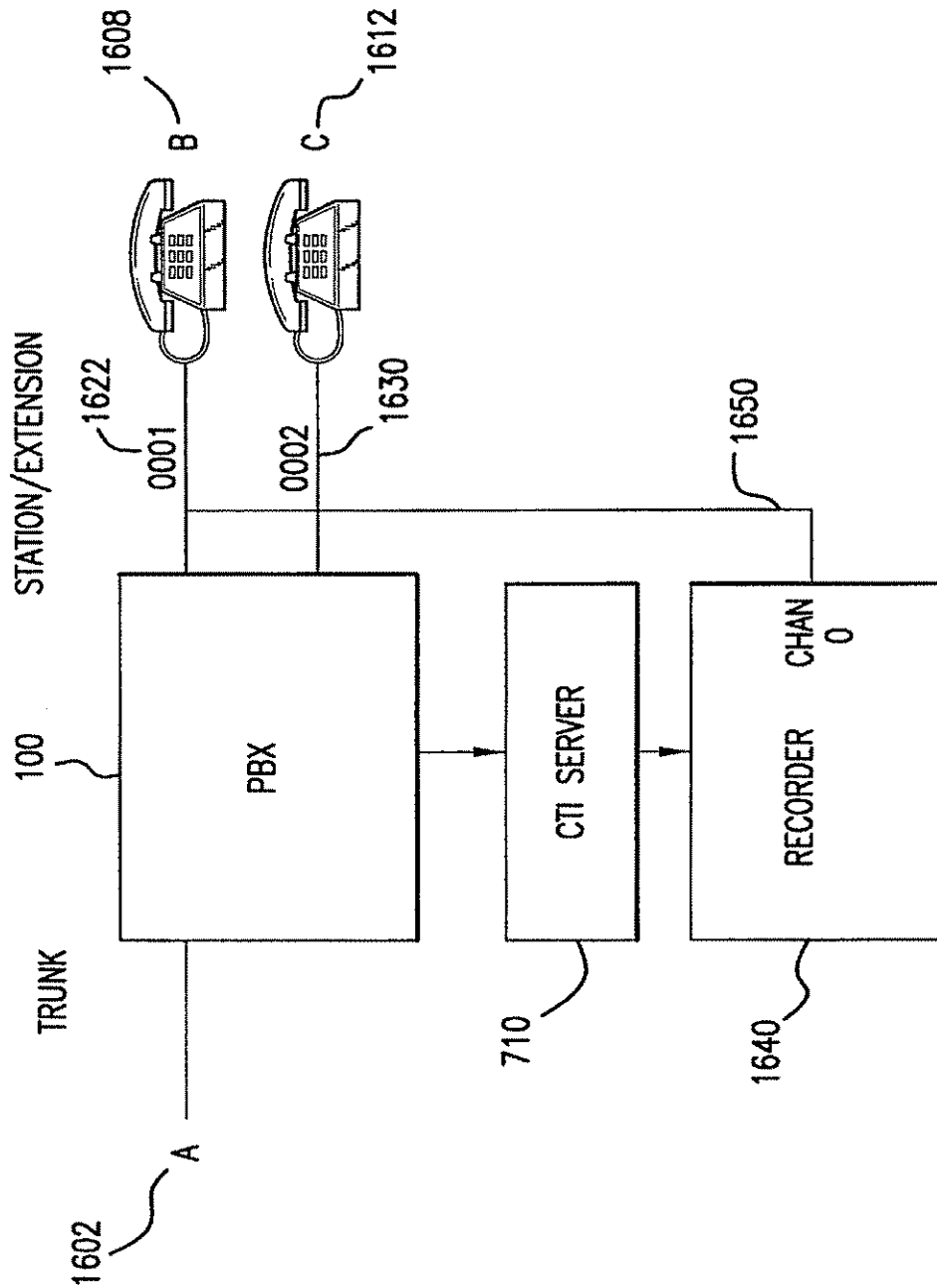


FIG.16A

U.S. Patent

Apr. 27, 2004

Sheet 18 of 29

US 6,728,345 B2

AgentId	EXTENSION	LOCATION	START TIME	END TIME	CONNECT REASON	DISCONNECT REASON
A		EXTERNAL	t_1	t_7	NORM START	NORM DROP
B	0001	INTERNAL	t_1	t_4	NORM START	Xfr AWAY
HOLD		INTERNAL	t_4	t_6	Xfr Rec	Xfr AWAY
C	0002	INTERNAL	t_6	t_7	Xfr Rec	OTHER PARTY HANGUP

FIG.16B

U.S. Patent

Apr. 27, 2004

Sheet 19 of 29

US 6,728,345 B2

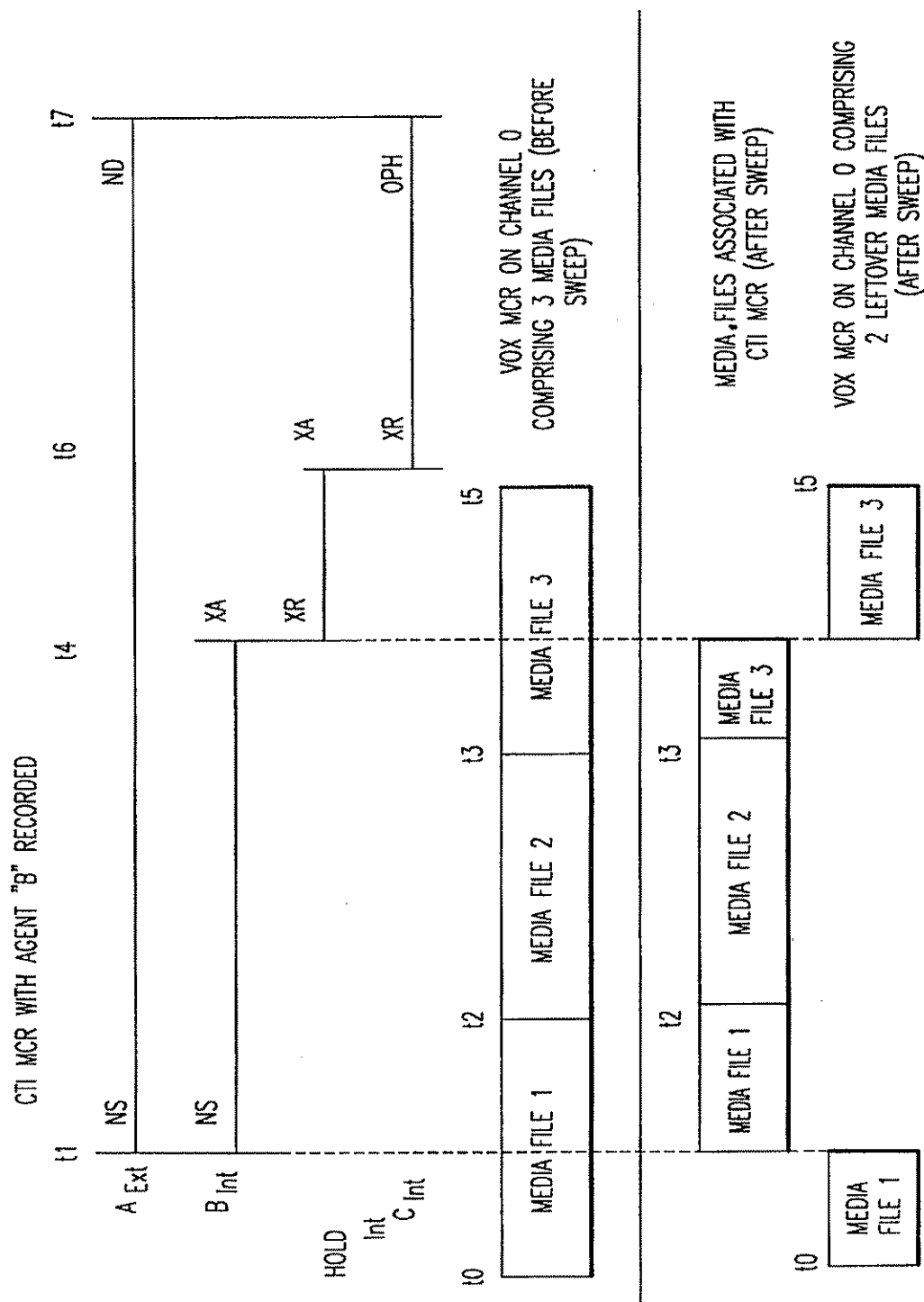


FIG.17

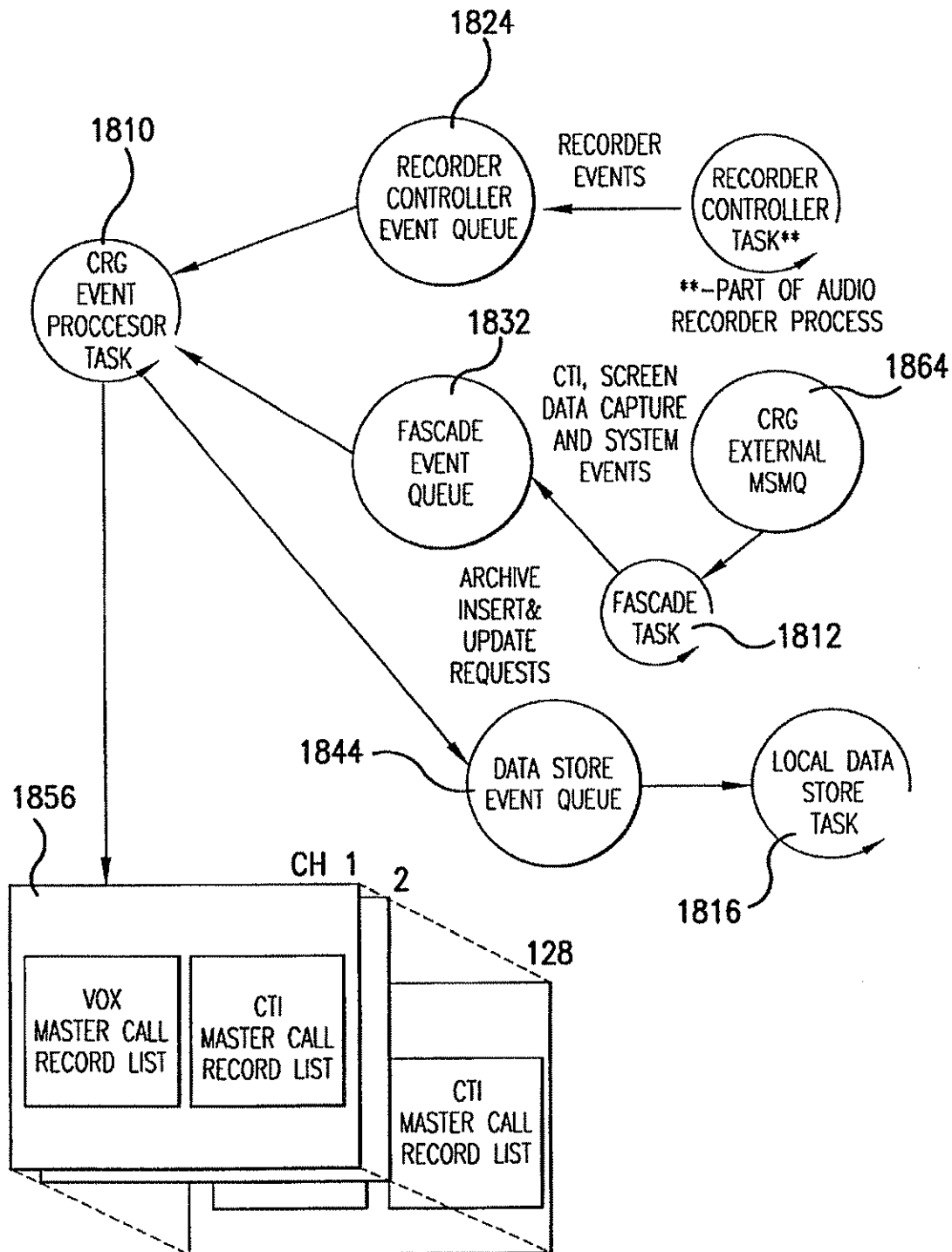


FIG. 18

U.S. Patent

Apr. 27, 2004

Sheet 21 of 29

US 6,728,345 B2

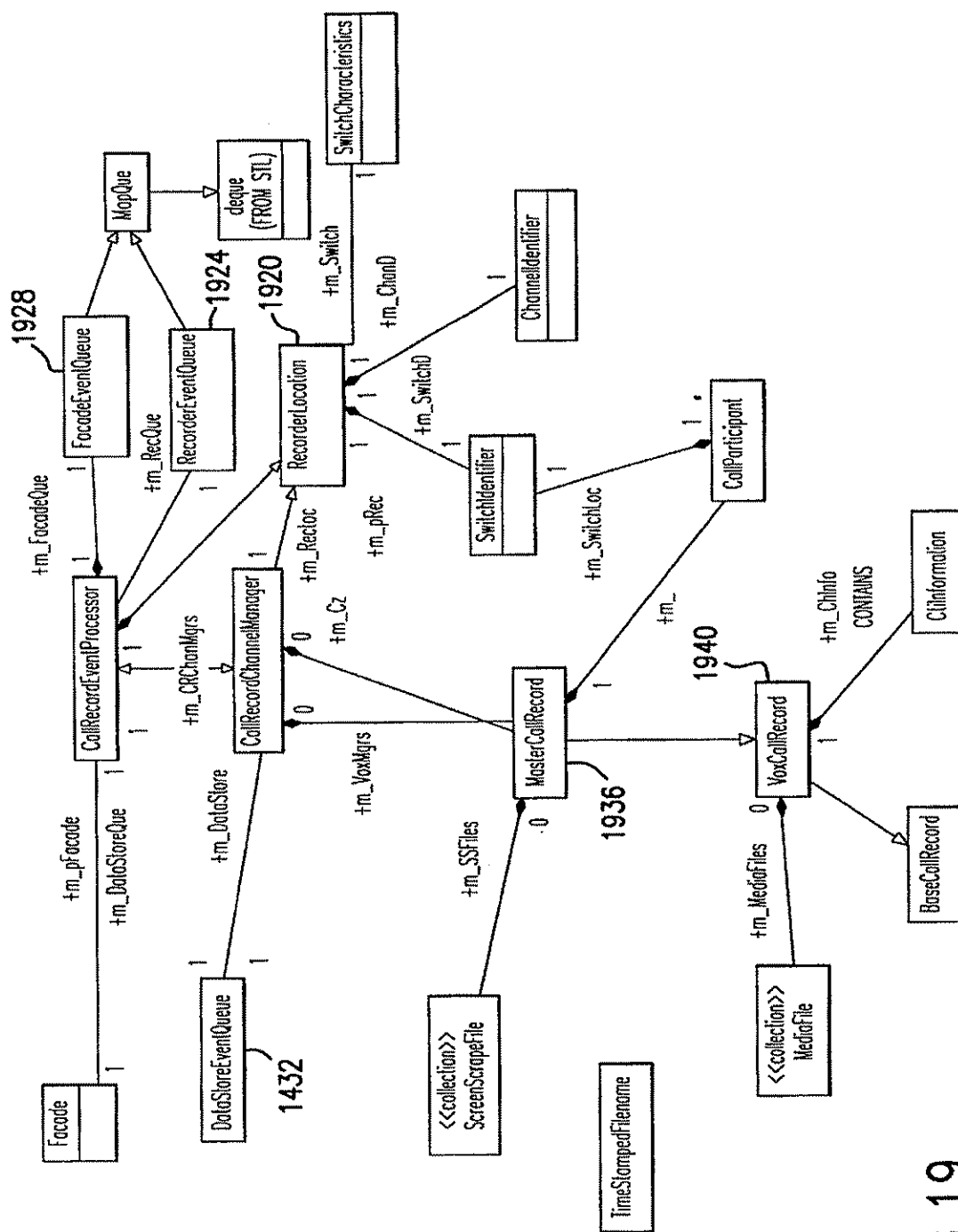


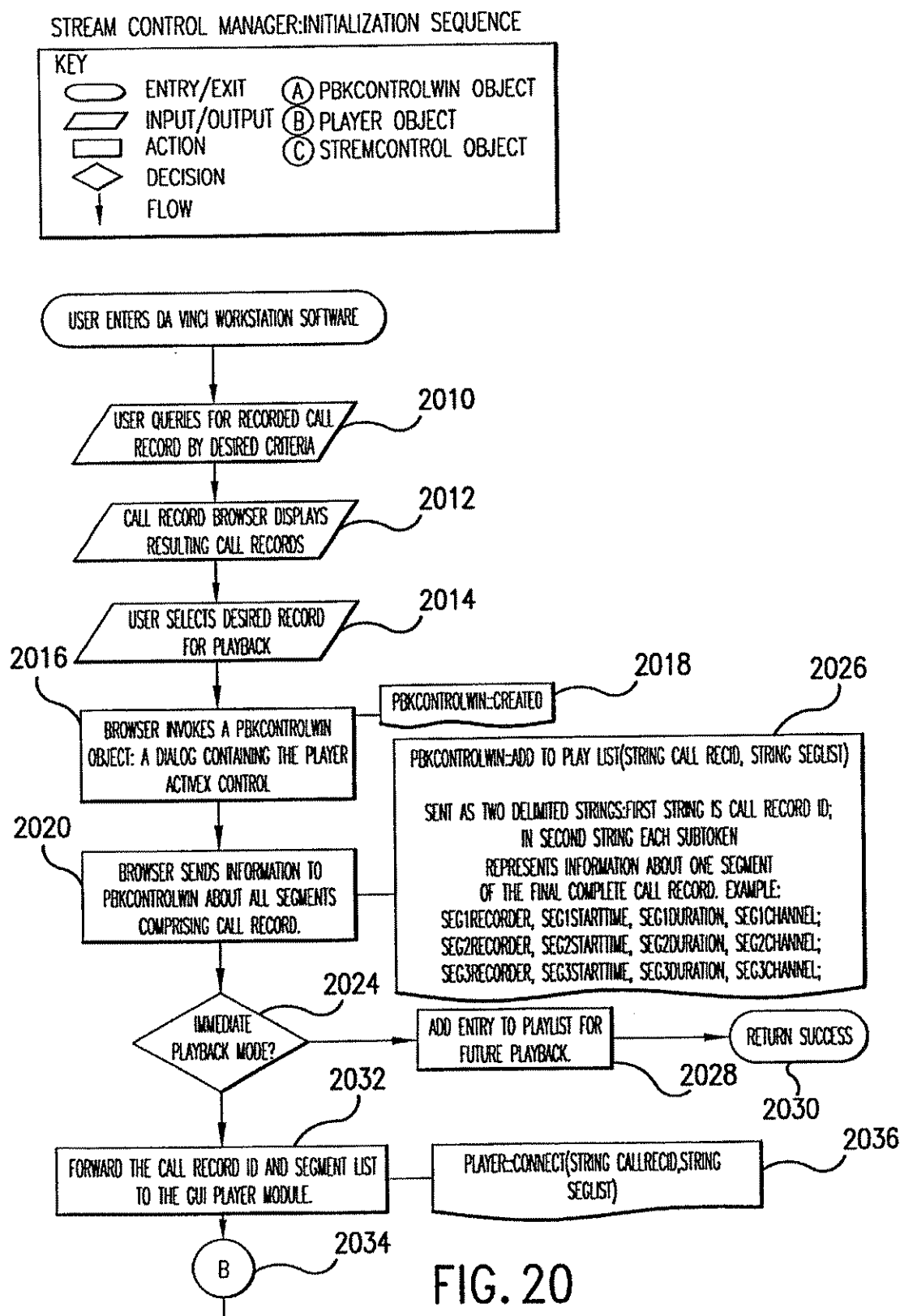
FIG. 19

U.S. Patent

Apr. 27, 2004

Sheet 22 of 29

US 6,728,345 B2



U.S. Patent

Apr. 27, 2004

Sheet 23 of 29

US 6,728,345 B2

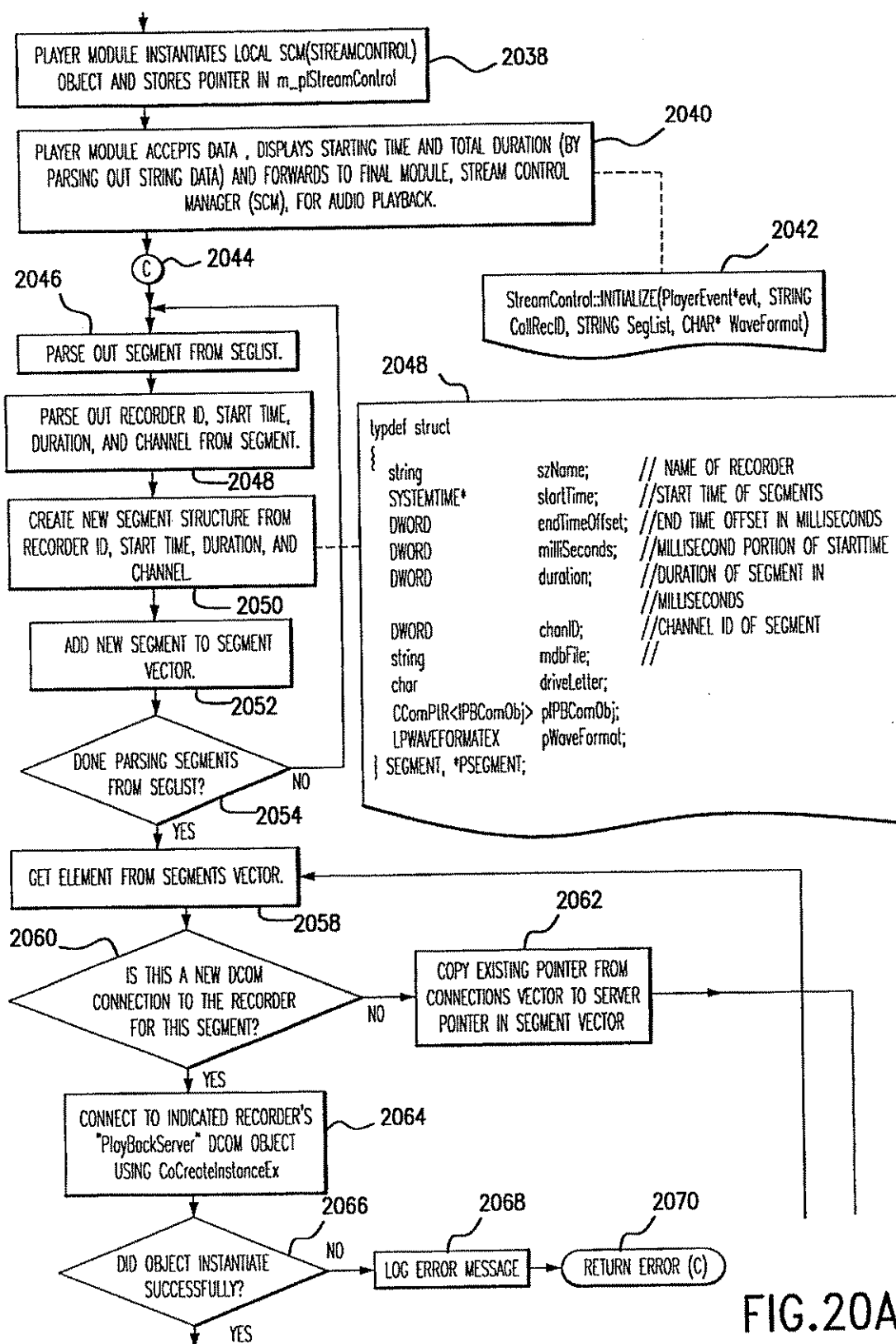


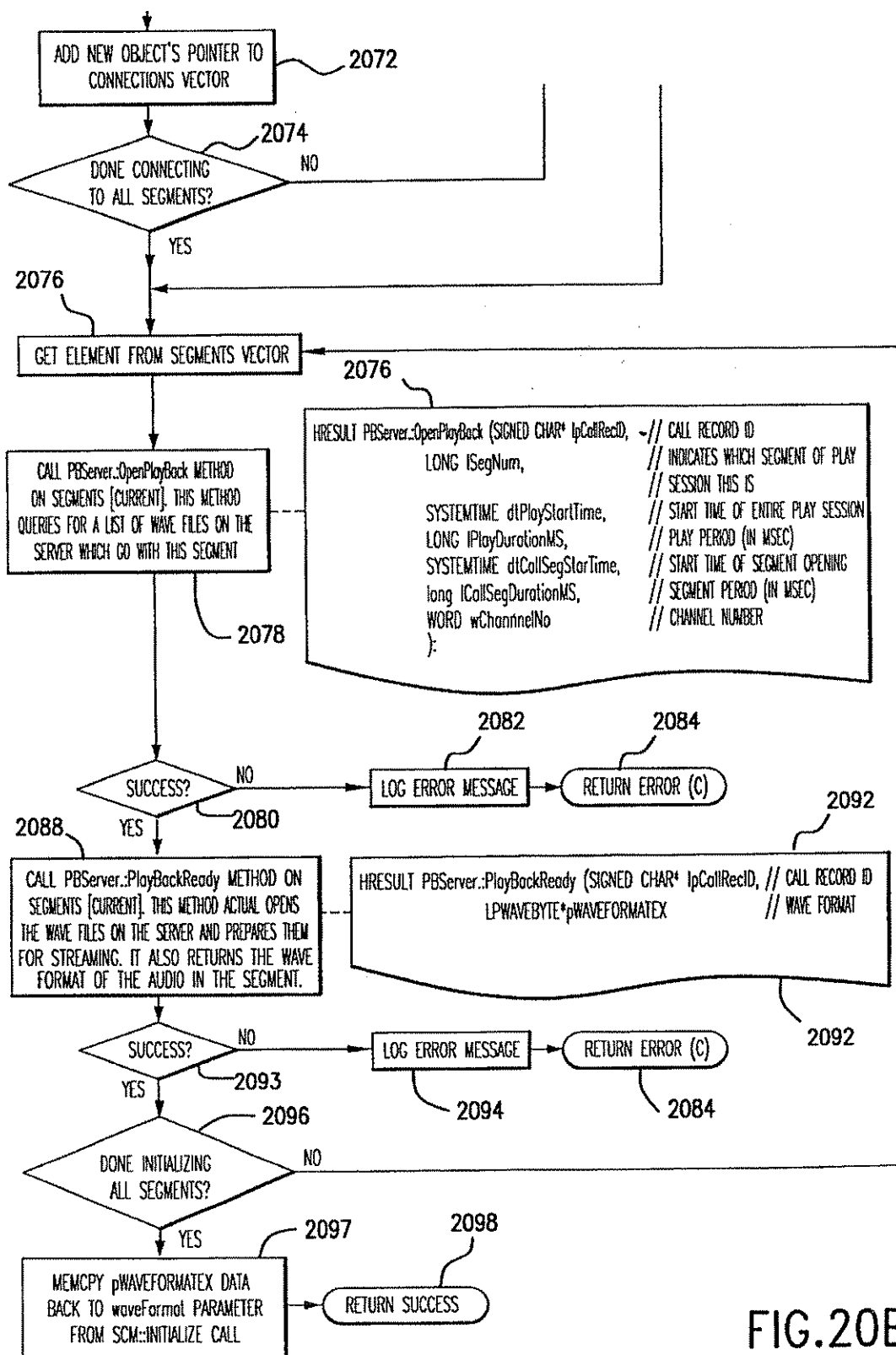
FIG. 20A

U.S. Patent

Apr. 27, 2004

Sheet 24 of 29

US 6,728,345 B2



U.S. Patent

Apr. 27, 2004

Sheet 25 of 29

US 6,728,345 B2

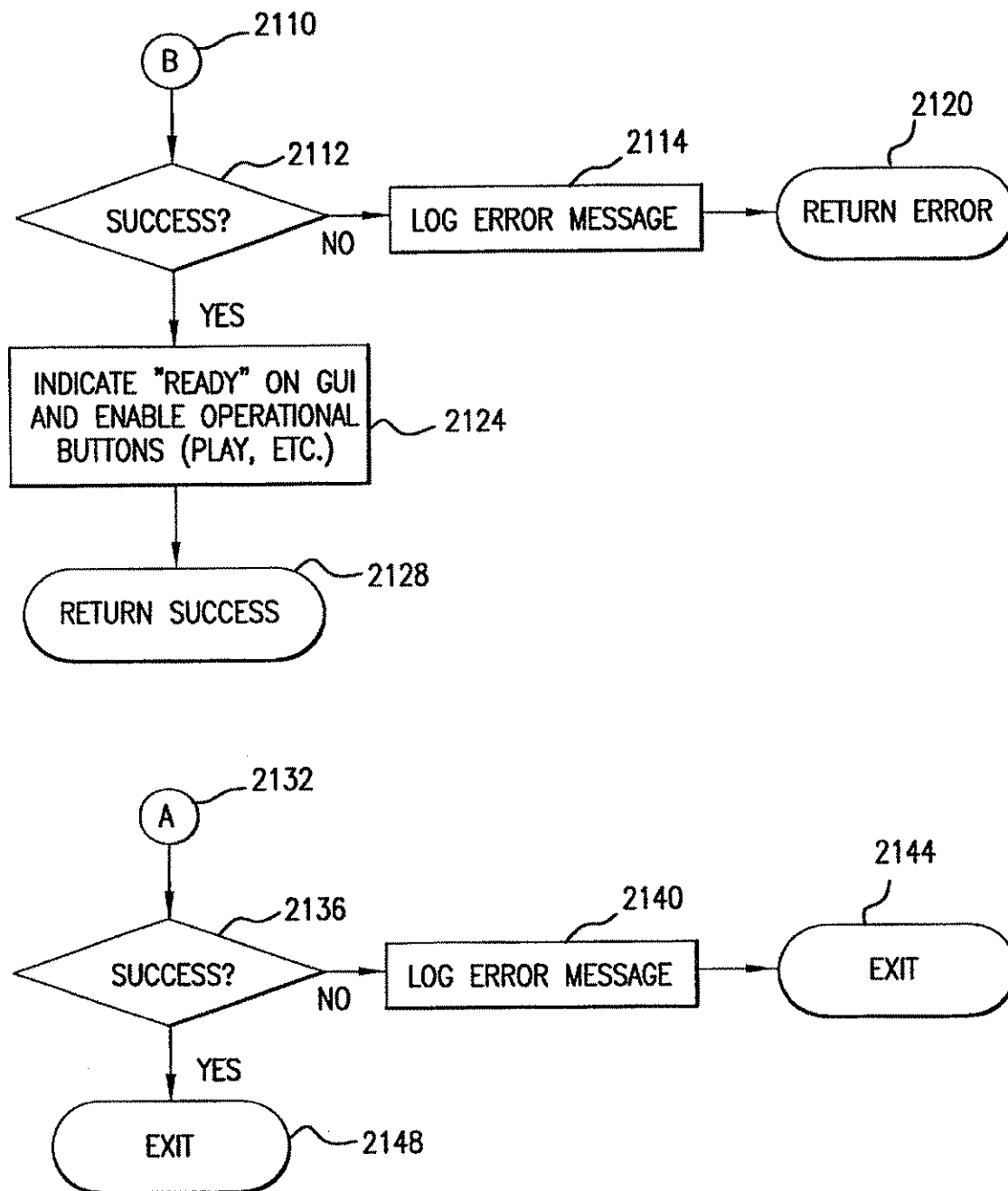


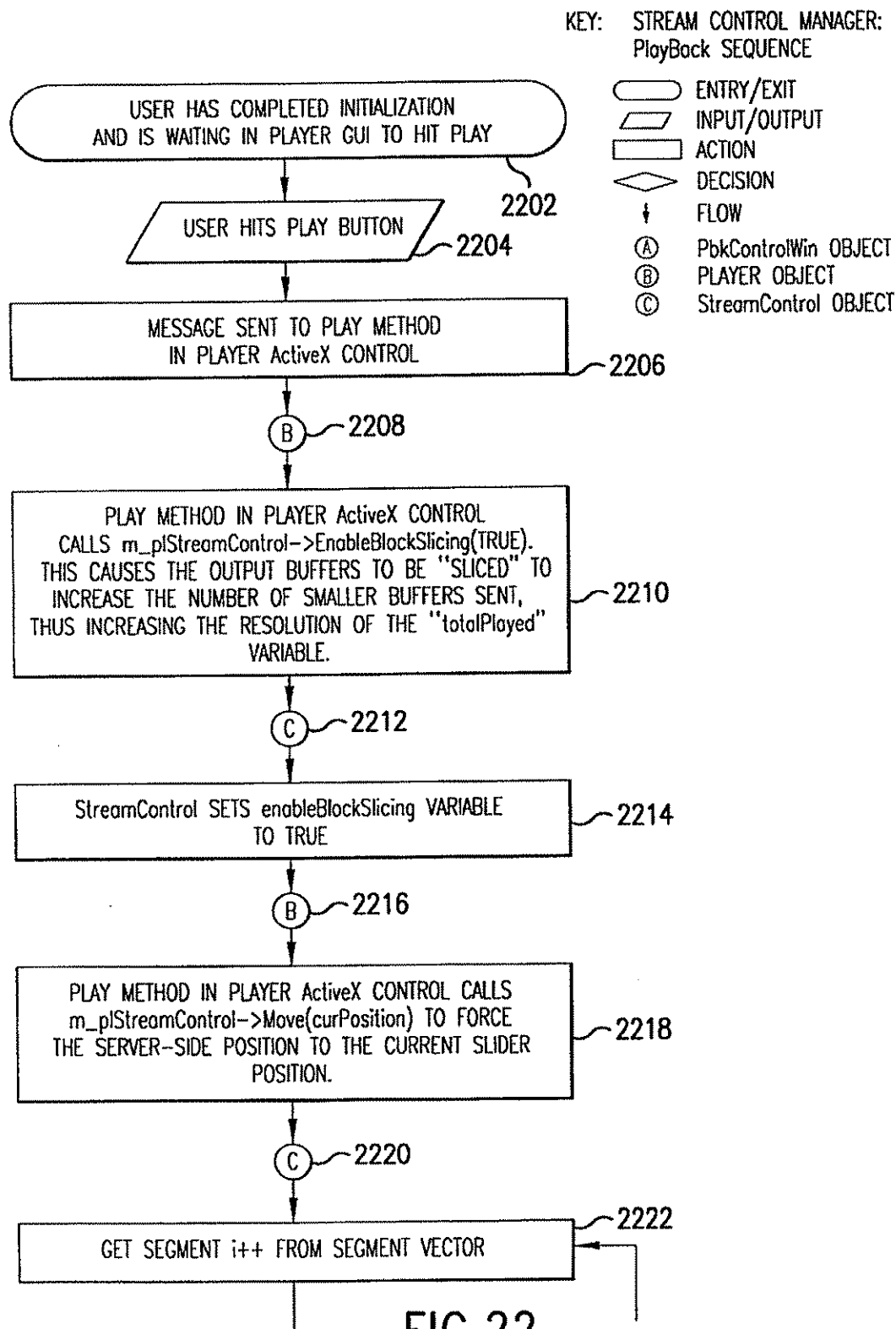
FIG.21

U.S. Patent

Apr. 27, 2004

Sheet 26 of 29

US 6,728,345 B2



U.S. Patent

Apr. 27, 2004

Sheet 27 of 29

US 6,728,345 B2

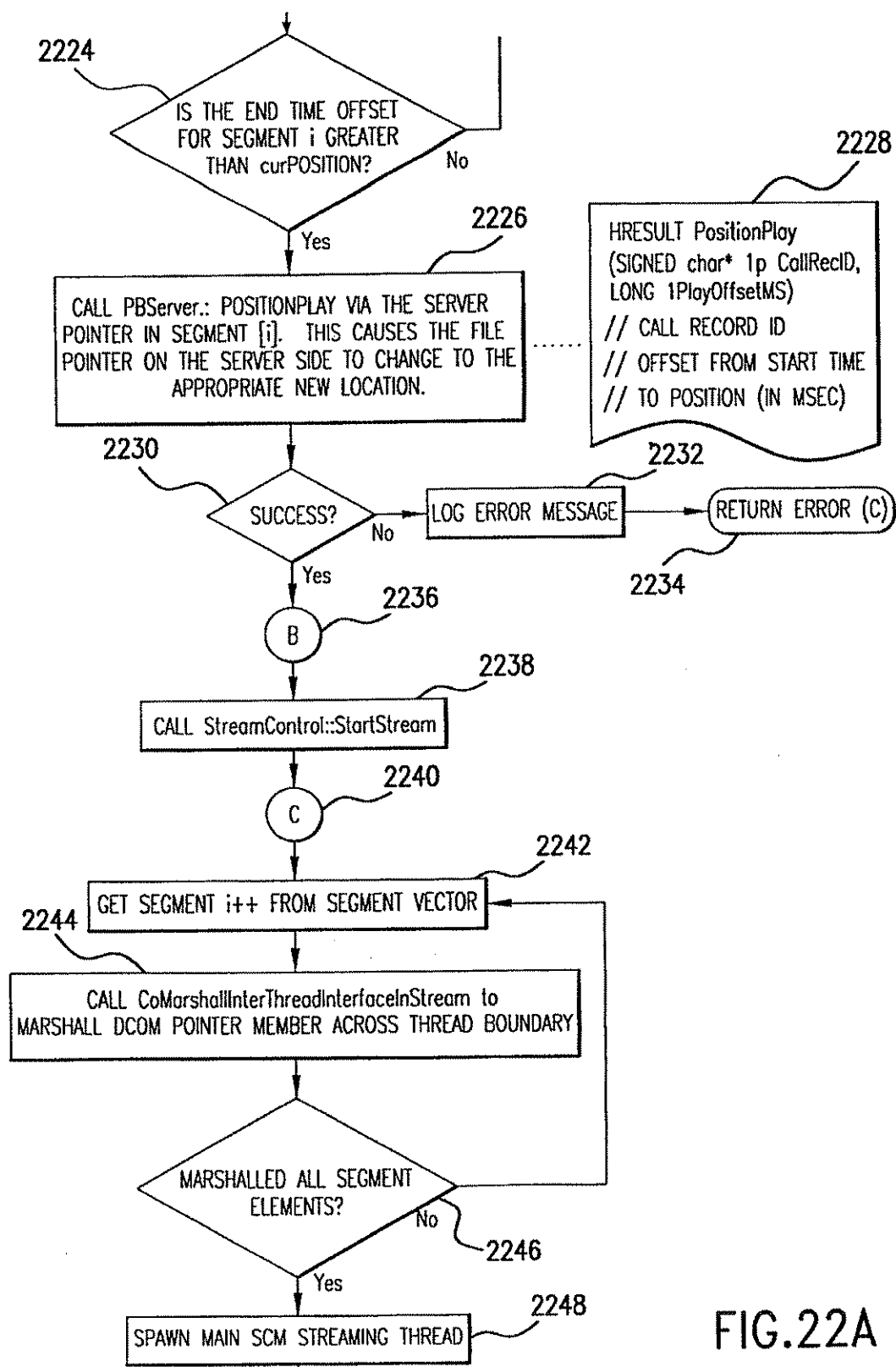


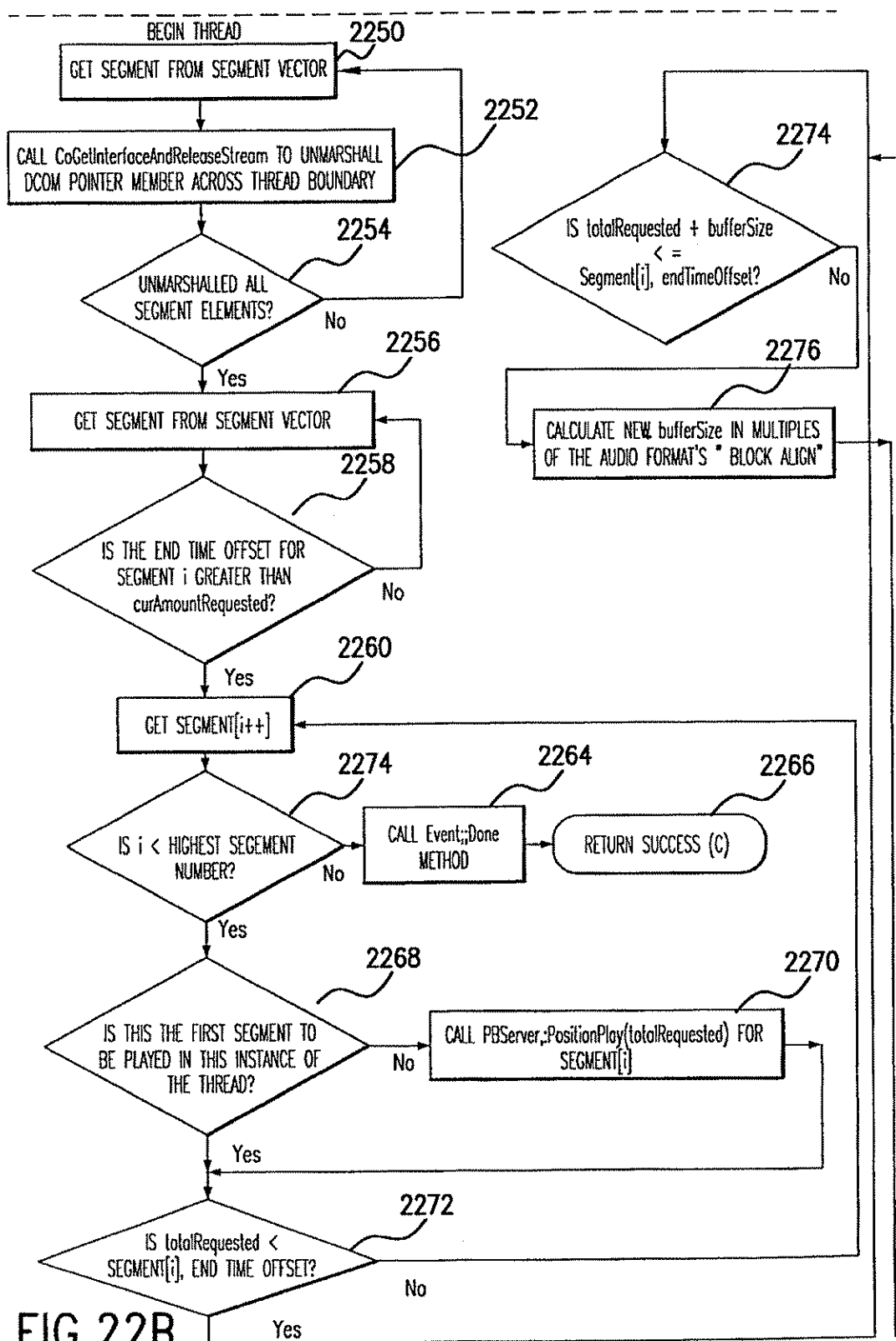
FIG.22A

U.S. Patent

Apr. 27, 2004

Sheet 28 of 29

US 6,728,345 B2



U.S. Patent

Apr. 27, 2004

Sheet 29 of 29

US 6,728,345 B2

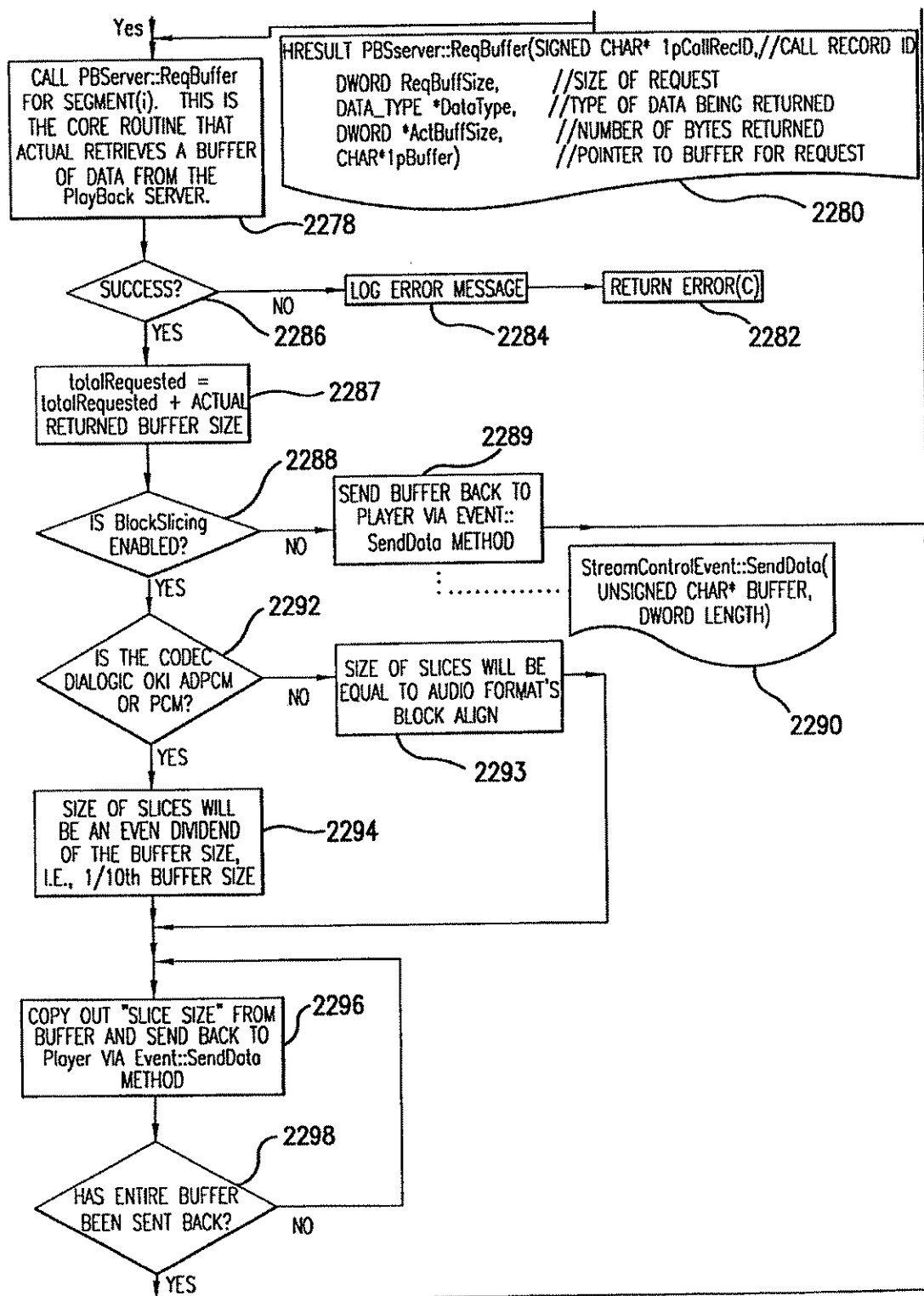


FIG.22C

US 6,728,345 B2

1

SYSTEM AND METHOD FOR RECORDING AND STORING TELEPHONE CALL INFORMATION

This is a continuation of application Ser. No. 09/328,299, 5
filed Jun. 8, 1999 now U.S. Pat. No. 6,249,570.

FIELD OF THE INVENTION

This invention relates generally to computer-aided data 10
recording. In particular, it relates to computer-aided monitoring and recording of telephone calls.

BACKGROUND OF THE INVENTION

Telephone call monitoring systems are used in a variety of 15
contexts, including emergency dispatch centers and commercial call centers. In many currently available call monitoring systems, a multitude of audio input sources ("channels") are monitored and recorded by a single hardware unit, and the audio recordings are saved and organized according to the input channel, date, time and duration. The capacity of the recording unit can be expanded to handle a larger number of channels by combining several recording units into a system using a local area network (LAN). Because retrieval is only possible using basic search criteria (recording unit, channel, date, time and duration), it is often 20
difficult to locate a particular audio recording that is of interest. When there is a need to search for a recording according to search criteria that are not directly supported by simple voice recording, locating a specific recording may require tedious and repetitive searching. For example, if there is a need to find a specific customer's call to resolve a disputed transaction, the recording unit or channel that carried the original call might not be known, so the searcher would be forced to manually play back many calls before finding the correct one.

With the advent of computer telephony integration (CTI), it is now possible to monitor a data link that supplies more information about telephone calls, in addition to simple voice recording. In a typical CTI system a telephone switch or private branch exchange (PBX) provides an interface suitable for processing by a computer, and expanded information about telephone calls is made available through this interface as the calls occur. Data fields that are available within this expanded information may include the external telephone number of the calling party, as well as identification numbers to help associate a series of events pertaining to the same call. With such a data link being used alongside a voice recording system, the search and retrieval system can be supplemented by constructing a database that combines the previously discussed basic search criteria with enhanced search criteria (based upon information obtained through a CTI data link) such as: telephone numbers of parties involved in the call; Caller ID (CLID) or Automatic Number Identification (ANI); Dialed Number Identification Service (DNIS); or the Agent ID Number of the Customer Service Representative.

As shown in FIG. 2, with suitable equipment for tapping into a voice communications line, a recording unit can intercept telephone call traffic using two methods. By attaching wires for recording channels on each extension within a call center, the traffic can be intercepted and recorded as it passes between the PBX and the agent telephone set. This first method is known as "station-side" recording 180. Alternatively, by attaching equipment on the trunk lines between the PBX and Public Switched Telephone Network (PSTN), the traffic can be intercepted at its point of entry

2

into the call center before the calls are dispatched by the PBX. This second method is known as "trunk-side" recording 170. Since businesses usually have more agent telephone sets than trunk lines, a "trunk-side" solution is likely to require less recording equipment and thus be less expensive. Another significant point for consideration is that "trunk-side" provides access only to external inbound or outbound calls, which are those typically involving customers of a business, whereas "station-side" also provides access to internal calls between agents (which may or may not relate to an external customer's transaction).

With respect to data links to provide call information to computers, there are typically two different categories of links from the PBX available. Some older links use interfaces such as SMDR (Station Message Detail Recording) or CDR (Call Detail Recording) that provide summary information about telephone calls in a line-oriented text format. Both acronyms refer to essentially the same type of system. Information from these links is generally provided after the call has concluded, and as such is suitable for billing applications or traffic analysis software. Many newer links use real-time interfaces that are designed to supply a series of events while a telephone call is still active within the PBX, to enable computer and multimedia systems to respond and interact with an external caller. The information provided by such real-time links is typically much more detailed than that provided by SMDR.

The detailed information and real-time nature of a CTI link is particularly important when building a recording system that is intended to react to telephone calls as they occur and to dynamically select which calls ought to be recorded or discarded. CTI-supplied information is also important when building a recording system that is intended to capture the full history of a telephone call, including recording the different agents who were involved in the conversation and how the call was held, transferred or conferenced. Likewise, real-time information is important in a system that intends to support (a) a live display of active calls, and (b) the capability for a user to listen and monitor the live audio traffic.

A "trunk-side" solution based upon voice recording alone will not satisfy the above requirements in a practical manner, since telephone calls are assigned to trunks dynamically as needed to handle the traffic. What trunk channel a particular call will be carried on cannot be predicted in advance. Without information to associate a logical telephone call with a physical recording of audio from a trunk channel, a user might have to search and retrieve many recordings before finding the one that is of interest. Moreover, in a system designed to make use of the enhanced search criteria provided by a data link, it would not be possible to programmatically associate the search data with the voice recording without information about the trunk channel where the call occurred.

This problem can be avoided as long as the data link provides sufficient information about the trunk channels being used for each call. Unfortunately, some PBX environments do not supply this critical information about trunk channels within the data provided on the real-time CTI link. For example, this problem is manifested by the Lucent Technologies DEFINITY G3 PBX, which is a commonly used telephone switch in North America. While the Lucent G3 PBX provides trunk channel information through its SMDR link, that information is not available until after the conclusion of the call. This presents a problem for system features and capabilities dependent upon real-time data. The real-time data link provided by the Lucent G3 PBX does not

US 6,728,345 B2

3

provide the necessary information about trunk channels. There is thus a need for a system which is capable of simultaneously monitoring both the SMDR link and the real-time CTI link, gathering information about calls from both sources, and combining that information into a single data model of the telephony activity within the call center. There is a further need for a system that combines the data model with information concerning the location of call recordings, resulting in a "master call record" that contains data matching each call with the segments of which it is comprised, and matching the data for each segment with the location of the recording of that segment. Such a system would facilitate monitoring, recording, and playing back complete telephone calls.

SUMMARY OF THE INVENTION

The present invention is directed to a system and method that is capable of simultaneously monitoring two or more data links, gathering information about calls from those data links, combining that information into a single data model of the telephony activity within the call center, and combining the data model with information concerning the location of call recordings, resulting in a "master call record" that contains data matching each call with the segments of which it is comprised, and matching the data for each segment with the location of the recording of that segment.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of the system of this invention in a preferred embodiment.

FIG. 2 illustrates the difference between trunk-side and station-side recording.

FIG. 3 shows a line-chart that illustrates various parties involved in a complex call.

FIG. 4 shows a schematic block diagram of a preferred embodiment for translating, summarizing and normalizing signals received from both an SMDR link and a Dialogic CT-Connect CTI service.

FIG. 5 illustrates the steps by which the translation module CtiCtc.exe integrates the data received from the CTI and SMDR links.

FIG. 6 illustrates how the CTI Server can be viewed as a set of logically distinct layers that deal with translating and distributing CTI events.

FIG. 7 illustrates how, in addition to telephony events, the CTI Server 710 is responsible for supplying certain metadata regarding agent events to the System Controller 130.

FIG. 8 shows the layout of the CTI Server.

FIG. 9 shows a version of CtiCtc.exe configured to work with a Lucent Telephony Services interface (and thus called CtiLts.exe instead of CtiCtc.exe).

FIG. 10 depicts key elements of the data model used in a preferred embodiment.

FIG. 11 illustrates three distinct layers of the CTI Server in a preferred embodiment.

FIG. 12 shows in block diagram form several threads of the CTI Server in a preferred embodiment that implement three distinct layers of processing (data collection, data normalization, and message emission).

FIG. 13 illustrates the program logic flow of the analyzer layer of the preferred embodiment.

FIG. 14 depicts the flow of information within the recording system of this invention in a preferred embodiment.

FIG. 15 shows how a recording unit operating with only voice signaling to guide the creation of its call records could make a number of fragmented audio segments.

4

FIG. 16 shows a graphical user interface used in the preferred embodiment.

FIG. 16A shows a system containing a CTI Server and a Recorder in a specific embodiment of the present invention.

FIG. 16B is a table illustrating descriptive information from the CTI Server used in a specific embodiment.

FIG. 17 illustrates steps in the creation of a Master Call Record used in a specific embodiment.

FIG. 18 shows the processing threads and data structures that comprise the CRG module in accordance with the present invention.

FIG. 19 illustrates the class diagram of the Call Record Generator used in a specific embodiment.

FIGS. 20, 20A, 20B, 21, 22, 22A, 22B, and 22C illustrate the operation of the Stream Control Manager.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The present invention is directed to a communication recording system and method. Generally, the functionality of the system involves tapping into activity on a PBX (Private Branch Exchange) by intercepting audio on either the trunk or station side of a telephone call. The tapped audio is then redirected as input to a channel on a Digital Signal Processor (DSP) based voice processing board, which in turn is digitized into program addressable buffers. The recorded digitized audio is then combined with descriptive information ("metadata") obtained through a Computer Telephony Integration (CTI) communications link with the PBX, and stored as a single manageable unit ("Voicedata") to facilitate its subsequent search and retrieval. The system uses modular architecture in both its hardware and software, so that any one component can be replaced or upgraded without affecting the rest of the system.

In a preferred embodiment the communications recording system comprises multiple rack-mountable computer-processing servers (such as the Compaq ProLiant 1600 R), using a multi-tasking operating system (e.g., Microsoft Windows NT), DSP voice processing boards (e.g., Dialogic D/160SC), and a distributed set of software components available from Dictaphone Corporation. In a specific embodiment directed to the smallest configuration, all of these components may reside in a single computer-processing server. In other preferred embodiments, related components are typically packaged in combinations and the entire system spans multiple servers that coordinate processing through a Local Area Network (LAN).

In this preferred configuration, the overall system generally comprises CTI Servers, Voice Servers, a Central Database Server, and User Workstations. CTI servers generally use a set of components to manage a data communications link with a telephone switch environment, to obtain notification of calls as they occur, along with the descriptive information about the calls (e.g., source and destination telephone numbers). The Voice Servers use a set of components to collect audio recordings, manage their storage, and facilitate their playback through the LAN. The Central Database Server uses a set of components to manage system-wide search and retrieval of recorded calls. User Workstations are typically desktop computers that use a set of components to allow a person to submit requests to search and retrieve recorded calls for playback and to control automatically scheduled functions within the recording system.

FIG. 1 shows in a block diagram from components of the system of this invention in a preferred embodiment. Data

US 6,728,345 B2

5

enters the recording system from a variety of sources. These sources can include a PBX 100, CTI middleware 105, ISDN lines 110, or other input sources 115. It will thus be appreciated that the system of the present invention can be used for monitoring and recording of information about any type of electronic communications. For simplicity, the following discussion uses the term Telephone calls. However, it is intended that term covers any electronic communication unless specified otherwise expressly.

Data from data sources 100, 105, 110 or 115 is transmitted to one or more CTI Translation Modules 165, which translates input data into a common format. The data is then sent to a CTI Message Router 120, which distributes the data onward to appropriate components of the system.

Audio Recorders 145 may be used for passive trunk-side 170 and extension-side 180 recording on a pre-determined static set of devices, as well as dynamically initiated recording of specific devices according to scheduling criteria through the Service Observance feature 185 provided by a telephone switch environment. The recordings are stored on an audio storage device 140. A Call Record Generator 150 matches data from the Audio Recorders 145 with data sent by the CTI Message Router 120 to create a Master Call Record (MCR) for each telephone call. The MCRs are stored in a Voicdata Storage module 155. One or more User Workstations 160 use the MCRs to reconstruct and play back complete or partial phone conversations stored in the audio storage device 140. A Scheduling and Control Services module 130 controls the Audio Recorders 145 and communicates with User Workstation 160. The Scheduling and Control Services module is responsible for starting and stopping the audio recording activity, according to pre-defined rules that are dependent upon time data provided by the Time Service 115 and CTI information. As the system components are packaged in the typical configuration, the CTI translation modules 165 and CTI message router 120 are co-resident upon a computer-processing server called the CTI Server 710. In a similar fashion, the combined set of components including the Time Service 125, Scheduling & Control Services 130, Audio Recorder 145, Audio Storage 140, and Call Record Generator 150, in a specific embodiment can be co-resident upon a computer-processing server called the Voice Server 124. The Voicdata storage 155 resides within a computer-processing server called the Central Database Server. The specialized application software for the User Workstation 160 resides upon desktop computers that use, in a preferred embodiment, Microsoft Windows 95, Windows 98 or Windows NT operating systems.

As noted above, in a specific embodiment the CTI Server comprises two main modules: a CTI translation module (such as the software program CtiCtc.exe, CtiLis.exe, and other translation modules) and a CTI Message Router module (such as the software program CtiServ.exe discussed below, or its equivalent). In a specific embodiment, the CTI Server may have several translation modules, for example, one for each PBX interface, or for each vendor API layer. As shown in FIG. 1, the CTI Server of the preferred embodiment accepts data from a PBX or similar equipment in a telephone switch environment, and can use both real-time CTI communications links and asynchronous information sources such the Station Message Detail Recording (SMDR) interface. The CTI Server translates and combines the various types of input data into a unified, normalized format. Normalized format information is then passed by the Message Router to various components of the system, as required.

As noted above, the Voice Server in a specific embodiment has several modules, including the Audio Recorder

6

145 and Call Record Generator (CRG) 150. The Audio Recorder collects a plurality of audio segments, representing the portions of a telephone call during which the sound level exceeded an adjustable tolerance threshold, thereby discerning alternating periods of speech and silence. Functionally, the Call Record Generator (CRG) produces Master Call Records, which encapsulate information (metadata) describing a telephone call. This descriptive information comes from a plurality of sources, including but not limited to an Audio Recorder and a CTI Server. The call records are created using a participant-oriented Call Record Model. The CRG then attempts to match the call records with existing recorded audio data. The CRG is thus able to combine data arriving in different chronological order into a single manageable entity which describes the complete history of a telephone call.

In a specific embodiment, a Playback Server (PBServer) (not shown) is a sub-component within the Audio Recorder module which uses call records to retrieve and play back telephone calls. Each recorder has its own PBServer, which is connected to a Player module (not shown) on the User Workstation 160. The Player module generally contains a Stream Control Manager module, which enables the Player module to use the PBServers to play back a telephone call which has several different participants and thus may have portions of the call stored on different recorders.

CTI Server

Still with reference to FIG. 1, when a call comes into the PBX system, both SMDR and real-time CTI data are generated by the PBX, and supplied to the recording system via the SMDR and CTI links. In accordance with the present invention, these two types of data are integrated by the CTI Server into a common format.

As known in the art, CTI (Computer Telephony Integration) supplements the recorded audio data in several important ways. CTI data is provided through a data communications link from specific telephone switching equipment located at the customer site. Supplied data comprises such items as the telephone numbers of involved parties, caller ID/ANI information, DNIS information, and agent ID numbers. ANI is Automatic Number Identification, a signaling method that identifies the telephone number of the calling party; the method is typically used by large-scale commercial call centers. DNIS is Dialed Number Identification Service, a feature that identifies the original "dialed digits," and that is commonly used in large-scale commercial call centers when multiple directory numbers are routed to the same receiving trunk group. In accordance with the present invention, the CTI server performs the task of analyzing and reorganizing data from both the real-time (CTI) and SMDR (asynchronous) links, and passing the results onwards into the recording system for further processing.

The design of the system of the preferred embodiment envisions that there will be a number of CTI translation modules 165 to accommodate a variety of possible input sources such as "native" PBX interfaces, CTI "middleware" vendors, ISDN data channel interfaces, etc. The system design incorporates flexibility in the manner in which CTI information is collected, making the system prepared to integrate with CTI links that may already exist at a customer site. The CTI Server of the preferred embodiment is capable of simultaneously monitoring both an SMDR link and a real-time CTI link, gathering information about calls from both sources, and combining that information into a single data model of the telephony activity within the call center.

US 6,728,345 B2

7

The CTI Server is responsible for supplying certain metadata regarding telephony events to the Voice Server's Call Record Generator 150. This metadata, such as called party and calling party numbers, trunk and channel ID, date and time, agent ID, etc., is combined by the Call Record Generator along with the other metadata, and data that is provided by the Audio Recorder 145 itself. Using this information, other components within the system are able to search for calls using a wide variety of useful and meaningful criteria, rather than simply using the recorder channel, date and time. As is known to those skilled in the art, an "event" is simply an action or occurrence detected by a computer program. The Call Record Generator 150 integrates that data into a single call record, which is updated after every event during the call, so that at the end of the call, the call's entire history is contained in the call record. The CRG matches the call record to the recording segments created by the Audio Recorders. The CRG integrates the call record with the metadata for the associated recordings of the phone call to generate a Master Call Record. When an operator wants to hear a recorded phone call, he uses the User Workstation (preferably equipped with a graphical user interface) to recall and play back the recorded call. Since the phone call may have had several different participants, pieces of the call may have been recorded on different recorders, each of which is associated with a different Playback Server. The system is nevertheless capable of playing back the entire phone call in the proper sequence.

In a preferred embodiment the CTI Server obtains the information regarding telephony events from various telephone switching environments, including PBXs, ACDs, and turret systems, which may have a wide variety of proprietary CTI interfaces. A telephone switching environment is a local telephone system that provides for routing of calls on a static or dynamic basis between specific destinations; the system is capable of identifying of when calls occur and who is involved in the calls. The CTI Server converts the information received into a common "normalized" format that is a simplified subset of the types of information available across the different vendors' PBXs, ACDs, and turret systems. This data conversion is partially facilitated by products such as Dialogic's CT-Connect API, which is capable of processing CTI messages from the major vendor's switches such as the Lucent DEFINITY G3, Nortel Meridian and DMS-100, Aspect, Rolm 9751, Rockwell Spectrum and Galaxy, Siemens Hicom, and Intecom. However, in accordance with the preferred embodiment an additional software layer exists within the CTI Server to further filter and normalize the CTI information. This feature also allows for a separate point of integration with customized software interfaces that may be necessary to connect with other switch vendors, especially certain turret systems that are not supported by Dialogic's CT-Connect (CTC) product. Alternate embodiments of the translation module use Lucent CentreVu Computer Telephony Server for Windows NT, or Genesys T-Server, as middleware instead of Dialogic CT-Connect. Additional alternate embodiments include direct "native" interfaces to a particular telephone switch, such as Aspect, without an interposing middleware product.

In terms of the CTI messages exchanged between the CTI Server and the various PBXs, ACDs, and turret systems, in accordance with a preferred embodiment the CTI Server is a "passive listener." That is, the CTI Server will monitor and receive information about call activity, but it will not send messages to affect, control, or redirect the calls. Using an "active" CTI server is also contemplated in specific embodiments.

8

Whereas the focal point of a Voice Server is recording content (e.g., audio clips), the metadata generated by the CTI Server is focused on describing the facts pertinent to the start and end points of each participant's involvement within a call. In other words, within the system of the preferred embodiment, recording is managed in a call-centric (rather than event-centric) fashion. This corresponds with the typical caller's point of view, in which a call is the entire conversation with a business entity, even if the conversation involves transfers to other agents or conferencing of multiple parties. The CTI Server generates events with metadata for the start and end points of the various recording segments of a complex conversation. These event records are interrelated by ID numbers and reason codes (see FIG. 3) so that the entire sequence of events for a complex conversation can be reconstructed by a browser application, preferably implemented on the User Workstation 160.

In accordance with the preferred embodiment, there can be one or more CTI Servers within the system of the subject system, as needed to process the traffic load of CTI information generated by multiple PBXs, ACDs, and turret systems. In a specific embodiment, a single CTI Server may be configured so as to connect with several PBXs, ACDs, and turret systems, depending upon the traffic load and physical connectivity requirements. In alternate embodiments, different CTI servers can be attached to different input sources. Generally, the number of CTI Servers within the system does not have a direct relationship with the number of Voice Servers. The telephony events generated by a CTI Server are individually filtered and re-transmitted to the appropriate Voice Server based upon configuration data for the system as a whole (managed by the Central Database Server), which maps the recording locations (extension number, or trunk & channel ID) with the Voice Server name and recording input port (channel).

During the active lifetime of a call, real-time information is accumulated within a historical call record that tracks each participant within the call. Each participant record includes descriptive fields for telephone numbers, agent ID numbers, time ranges, and reason codes for joining and leaving the conversation. At certain key points during the accumulation of data, whenever a party joins or leaves the conversation, the call record is transmitted onward to allow the rest of the recording system to process the information accumulated thus far. Upon the conclusion of the call, the CTI server retains a copy of the call record for a configurable time interval before discarding it from memory. This delay is intended to allow for the arrival of the SMDR data.

Upon receiving SMDR data, the CTI server searches its memory for a call record pertaining to the same logical telephone call that would have been accumulated from previous real-time messages. Matching this information is not a trivial task, since the SMDR link and real-time CTI link do not share a common reference ID number for use by their messages in describing the occurrence of the telephone call.

Therefore the software of the preferred embodiment must use other "clues" to guide the matching process, by comparison on a combination of other data fields that exists in common between the SMDR and real-time CTI data. These data fields include: (1) the telephone number of the first external party involved in the call; (2) the telephone number of the first internal party involved in the call; (3) the direction of the call (e.g., inbound, outbound); (4) the start time of the call, in hours and minutes; and (5) the duration, in seconds, of the call.

Once again, the matching process is not trivial because the SMDR link gives the starting time of the call only in hours

US 6,728,345 B2

9

and minutes, whereas the starting time given by the real-time link also includes seconds. It is quite possible that more than one call could be started and stopped within a single minute. This would result in an ambiguous match, if not combined with other search fields. The same argument holds true for each of the other fields upon which a match can be performed. No single field alone will provide an unambiguous matching of the records. Even in combination, it is conceivable (although statistically unlikely) that an ambiguous case could occur: if the same two parties were to call each other twice within the span of a minute, and each call was roughly the same length in seconds. The odds of such a problem are increased if a large number of calls are routed through a common entry point into the call center, as would be the case if the first internal party involved in the call is a shared Voice Response Unit (VRU) or Automatic Call Distribution (ACD) queue. In addition, if information about the external party's number is missing due to limitations of the PSTN or incoming trunk configuration, matching the call records becomes even more problematic.

Adding to these difficulties is the fact that clock-time values reported by the SMDR link and the real-time CTI link may not be perfectly in synchronization with each other. Therefore, the preferred embodiment comprises a mechanism in which an imperfect match of times can be tolerated, while still retaining an acceptable level of reliability in matching the call records.

Because these various factors require a degree of flexibility in the matching algorithm, the preferred embodiment incorporates a weighted formula that is applied to potential match candidates. The formula yields a numerical confidence factor that can be used to select the best apparent match candidate. For each of the "clues," a test is conducted to determine the quality of matching on that data field. This matching quality is rated as a percentage. Certain fields, such as time values, are allowed to vary within a configurable tolerance range, whereas other fields are required to match exactly or not at all. After the matching quality of a field has been determined, it is multiplied by an importance factor that applies a relative weight to each of the various fields that can be examined during matching. The final confidence factor is the summation of these calculations:

$$\text{Confidence Factor} = \sum_i ((\text{Match Quality})_i * (\text{Weighting Factor})_i)$$

In order to account for the fact that characteristics of the call traffic may vary significantly between individual call centers, the tolerance factors (e.g., for time value offsets) and the weighting factors are re-configurable. There is also a re-configurable minimum level for confidence factors, below which the match candidate will always be rejected.

For those fields, such as time or duration, where an imprecise match may be allowed, the configuration data will define an allowable variance range (plus or minus a certain number of seconds). Values that do not match exactly, but fall within the variance range, are rated with match quality expressed in percentage that is measured by one minus the ratio of the difference from the expected value versus the maximum variance.

$$\text{Match Quality} = 1 - (\text{abs}(\text{Expected Value} - \text{Actual Value}) / \text{Maximum Variance})$$

Values outside the variance range are rated as a match quality of zero. This produces a linearly scaled match quality. Alternate embodiments may use other distributions (e.g., standard deviation "bell curves") to produce a non-linear scale for the match quality. Where an exact match is required for a field, the match quality is either 100% or zero.

10

EXAMPLE

Real-time CTI events report a telephone call from an unknown external party (sing or deliberately suppressed ANI/CLID information) to an internal party at extension 1234, starting at 12:25:03 and lasting for 17 seconds (CLID is Calling Line Identification, a signaling method that identifies the telephone number of the calling party; the method is typically used by residential subscribers and small businesses). Two SMDR records arrive which could possibly match with this call. The first record indicates an inbound call received by extension 1234 at 12:26 and lasting 26 seconds. The second record indicates an inbound received by extension 1234 at 12:27 and lasting 20 seconds. The system is configured with a variance range of plus or minus 3 minutes for the start time, and plus or minus 10 seconds for the duration.

Weighting Factors are:

20	External Party Telephone Number
40	Internal Party Telephone Number
30	Direction
20	Start Time
20	Duration

Confidence Factors are therefore calculated as follows:

$$CF_1 = (20 * 1.00) + (40 * 1.00) + (30 * 1.00) + (20 * (1 - 1/5)) + (20 * (1 - 3/10)) = 105 \frac{1}{5}$$

$$CF_2 = (20 * 1.00) + (40 * 1.00) + (30 * 1.00) + (20 * (1 - 1/4)) + (20 * (1 - 3/10)) = 110 \frac{3}{10}$$

system will therefore match the CTI events with the second SMDR record.

After a match has been selected, the trunk channel information (and any other useful information that can supplement the previously gathered real-time CTI data) is extracted from the SMDR data and added to the call record within the CTI server's data model of telephony activity. Then the updated call record is transmitted onward to allow the rest of the recording system to process it. With the trunk channel information at hand, the recording system is able to associate the enhanced logical search information with the physical voice recording, and take whatever actions may have been dependent upon this information, such as selectively recording or discarding the call.

FIG. 2 is an illustration of the difference between trunk-side and station-side recording at a call center with agents. With suitable equipment for tapping into a voice communications line, a recording unit can intercept telephone call traffic using either of these two methods. By attaching wires for recording channels 180 on each extension within a call center, the traffic can be intercepted and recorded as it passes between the PBX 100 and the agent telephone sets 230. This first method is known as "station-side" recording. Alternatively, by attaching equipment 170 on the trunk lines between the PBX and Public Switched Telephone Network (PSTN) 250, the traffic can be intercepted at its point of entry into the call center before the calls are dispatched by the PBX. This second method is known as "trunk-side" recording. Since businesses usually have more agent telephone sets than trunk lines, a "trunk-side" solution is likely to require less recording equipment and thus be less expensive. Another significant point for consideration is that "trunk-side" provides access only to external inbound or outbound calls, which are those typically involving customers of a business, whereas "station-side" also provides access to

US 6,728,345 B2

11

internal calls between agents (which may or may not relate to an external customer's transaction).

A third type of recording interface is Service Observance 185 (see FIG. 1), which is physically wired in manner like station-side recording, but using separated dedicated lines to a recording input channel rather than being interposed between a PBX and telephone set. In this mode of operation, the Recorder joins into a telephone call as a silent conference participant using the PBX Service Observance feature (originally intended to enable a supervisor to directly monitor an employee's telephone calls upon demand). This differs from ordinary station-side recording in that the internal party being recorded on a given input channel can vary upon demand rather than being fixed by the wiring pattern.

FIG. 3 shows a line-chart that illustrates various parties involved in a complex call. A is the customer phone number, and B and C are the agent phone numbers located behind recording channels R20 and R21 respectively (see FIG. 2).

Initially, the call comes in from line A 335 to line B 340. A real-time CTI message occurs describing that phone B is ringing, but not yet answered. B answers the phone 365 at time t0 310. The "NS" at 360 indicates the normal start of a phone call. A real-time CTI message occurs describing the start of the call between A and B. The telephony model is updated to reflect the fact that the call between the initial 2 participants (A and B) started normally at time t0 310. A copy of the call record is then sent onward to the rest of the recording system. The call record is retained within the telephony model, associated with device (or line) B. At time t1 315, B places the call on hold 370 (the "XA" at 370 indicates that the call was transferred away from B; the "XR" at 375 indicates that the transfer was received by HOLD). A real time CTI message occurs describing that B placed the call on hold. The telephony model is updated to reflect that B transferred the call to HOLD 345 at time t1 315. (This information is accumulated with the information previously gathered at t0 310). A copy of the call record is then sent onward to the rest of the recording system. The call record is removed from device B within the telephony model, but kept in a list of held calls.

At time t2 320, B returns to the call 380 and conferences in C 355 (the "XA" at 380 indicates that the call was transferred away from HOLD; the "XR" at 382 indicates that the transfer was received by B; the "CA" at 384 indicates that C was added as a conference participant). A real-time CTI message occurs describing that B returned to the call and invited C by conferencing. The call record is moved within the telephony model from the list of held calls back to device B. The telephony model is updated to reflect that HOLD 345 transferred the call 380 back to B at t2 320. (Note that information is accumulated with the information previously gathered at t0 310 and t1 315). A copy of the call record is then sent onward to the rest of the recording system. The telephony model is updated to reflect that C joined the call 384 as a conference participant at t2. (This information continues to be accumulated with previously gathered information). A copy of the call record is then sent onward to the rest of the recording system. The call record is retained with both devices B and C within the telephony model.

At time t3 325, a real-time CTI message occurs describing that C dropped out 386 of the call (the "CD" at 386 indicates that C was dropped from the conference). The telephony model is updated to reflect that C dropped out of the conference at t3. (This information continues to be accumulated with previously gathered information). A copy of the

12

call record is sent onward to the rest of the recording system. The call record is removed from device C within the telephony model, but retained with device B.

At time t4 330, A terminates the call to B. A real-time CTI message occurs describing that A terminated the call (The "ND" at 390 indicates that a normal drop of the call occurred; the "OPH" at 395 indicates that the other party hung up). The telephony model is updated to reflect that A stopped normally and B stopped because the other party hung up at t4. (This information continues to be accumulated with previously gathered information). A copy of the call record is then sent onward to the rest of the recording system. The call record is then removed from device B, but kept in a list of completed calls. An SMDR message is received which summarizes the call in its entirety. The list of completed calls is searched to find a match, and the appropriate call record is retrieved. The call record is updated with the trunk channel information from the SMDR message. A copy of the call record is sent onward to the rest of the recording system. The call record is removed from the list of completed calls.

FIG. 4 shows a schematic block diagram of a preferred embodiment for translating, summarizing and normalizing signals received from both an SMDR link and a Dialogic CT-Connect CTI unit. In the embodiment illustrated in FIG. 4, the recording system of the subject system is represented by daVinci™, a new generation recording system of Dictaphone Corp. Alternatively (or simultaneously), Dictaphone's Symphony™ CTI software can be used, in conjunction with Dictaphone's ProLog™ recording system (the system preceding daVinci™). Hereinafter, the translation/summarization module of the preferred embodiment illustrated in FIG. 4 will be referred to as CtiCtc.exe.

The module CtiCtc.exe is itself comprised of a plurality of modules, as shown in FIG. 4. A CtiAgentEvent module 448 is comprised of a data structure for agent log-on and log-off messages. A CtiAgentStatusFile module 454 manages a file that tracks agents currently logged on. A CtiCallEvent module 416 is comprised of a data structure for a call record (i.e., normalized and summarized CTI events). A CtiCallState module 418 is comprised of a generic data structure to represent the state of telephony activity at a particular location (extension, hold area, etc.). A CtiComMessageEmitter module 476 comprises a layer that converts the stream of CtiCallEvent objects (generated by a CtiCtcAnalyzer 456) into a format that can be sent to other da Vinci system components. A CtiCtcAnalyzer module 456 comprises a processing engine which examines CTC and SMDR messages and keeps track of a state machine for the activity on each extension. The CtiCtcAnalyzer module performs normalization of the CTC and SMDR data.

A CtiCtcAnalyzerUtils module 452 comprises a collection of utility subroutines that assist in examining the CTC and SMDR messages. A CtiCtcCallState module 420 comprises a data structure that represents the state of telephony activity at a particular location (extension, hold area, etc.) including CTC-specific information. A CtiCtcCallStateList module 432 manages an open-ended collection of CtiCtcCallState objects. This collection of objects is typically used to track calls that are "held" or "bumped." A CtiCtcData module 428 comprises a data structure wrapped around the raw CTC data, with the addition of a time stamp indicating when a message arrives. A CtiCtcDataFile module 412 manages a file of CtiCtcData objects that can be captured or displayed. A CtiCtcExtensionInfo module 442 manages a collection of CtiCtcCallState objects, with one object for each extension.

US 6,728,345 B2

13

A CtiCtcInput module 464 comprises an input source engine that obtains incoming CtiCtcData objects, either from a "live" server or from a playback file. A CtiCtcMain module comprises the main() function for CtiCtc.exe. The main() function handles command line and registry parameters, along with other start-up processing. A CtiCtcParameters module 472 comprises data structure and program logic for managing the configuration parameters in the Windows NT registry. A CtiCtcScanner module 446 comprises a utility module for building a list of all available extensions on a particular telephone switch. A CtiCtcStats module 434 comprises a data structure for compiling statistics on the number of CTC, SMDR, and CTI messages. A CtiDtpField module (not shown) is used by a CtiDtpMessageEmitter module 478, and comprises a data structure for an individual field in the Dictaphone Telephony Protocol ("DTP"), used to communicate with other Symphony CTI system components. A CtiDtpMessage module (not shown) is used by a CtiDtpMessageEmitter module 478, and comprises a data structure for a complete message in the DTP to be sent onwards to the Symphony CTI system.

A CtiDtpMessageEmitter module 478 comprises a layer that converts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) into a format that can be sent to the Symphony CTI recording platform. A CtiDtpSocketSrv module (not shown) manages the TCP/IP connection through which messages for DTP are sent to the Symphony CTI platform. A CtiDtpUtility module (not shown) comprises a collection of utility routines that assist in examining and processing DTP messages. A CtiExtensionFile module 450 manages the configuration file that lists all available telephone extensions. A CtiExtensionInfo module 440 manages a collection of CtiCallState objects, with one object for each extension. A CtiExtensionNumber module 430 comprises an abstraction of an individual extension number as either a numerical or string value, so that changes to this model will not have a global impact in CtiCtc.exe.

A CtiMessageEmitter module 458 comprises an abstract layer that converts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) into a format that can be sent to various target platforms, including the da Vinci and SymphonyCTI systems. A CtiMessageEmitterParameters module 474 comprises a data structure and program logic for managing configuration parameters that relate only to the message emitter(s). A CtiMessageQueue module 462 comprises shared memory for transferring data between threads. As is known to those skilled in the art, a "thread" is a part of a program that can execute independently of other parts. A CtiNullMessageEmitter module 460 comprises a layer that accepts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) and discards them instead of sending them to a target platform. Typically this layer is used only when debugging CtiCtc.exe, or to capture a sample file of CTI events from a PBX without sending them to the da Vinci or SymphonyCTI systems. A CtiPartyListElement module 414 comprises a sub-component of the CtiCallEvent data structure 416. The module 414 tracks information about an individual participant (e.g., caller, recipient) in a call.

A CtiPeriodicMsg module 468 comprises a generic handler for sending timer-based housekeeping messages. A CtiPrint module 444 comprises a layer that manages console output and conditional trace messages. A CtiSmdrData module 424 comprises a data structure wrapped around the raw SMDR data, with the addition of a time stamp indicating when a message arrives. A CtiSmdrDataFile module 408 manages a file of CtiSmdrData objects that can be captured

14

or replayed. A CtiSmdrDataList module 422 manages an open-ended collection of CtiSmdrData objects. This is typically used to buffer SMDR records that have not been paired with CTC records. A CtiSmdrInput module 466 comprises an input source engine that obtains incoming CtiSmdrData objects, either from a "live" server or from a playback file.

A CtiTagNames module 436 comprises a utility module that converts number values to descriptive strings for debugging and tracing purposes. A CtiTime module 438 comprises a utility module that converts time values to UTC for internal storage and conditionally prints times in either the UTC or local time zone. A CtiTrunkMap module 426 comprises a data structure that describes a mapping between logical trunks and logical trunk groups, into physical trunks and TDM timeslots. A CtiTrunkMapFile module 410 manages a configuration file that contains the CtiTrunkMap information.

FIG. 5 illustrates the steps by which the translation module CtiCtc.exe integrates the data received from the CTI and SMDR links. Initially, at step 502, the translation module receives a message from the SMDR link or from the CTI link. If the message is determined, at step 504, to be a CTI message, the current data model of telephony activity is updated at step 506. If the translation module determines at step 514 that the CTI message indicates a party joined or left the call, the call record is at step 518 transmitted onward to the rest of the recording system before continuing to step 512. Otherwise, no message is transmitted onward to the rest of the recording system and processing continues directly to step 512. If the translation module determines at step 512 that the CTI message indicates that the call has been concluded, at step 520 the module removes the call record from the associated devices. The translation module then adds the call record to the list of recently completed calls at step 528. Completed calls are discarded (step 530) after they get too old (i.e., after a predetermined number of recorded calls, or a given time period after the original recording of the call). Processing then continues again from step 502 by receiving the next incoming message. If at step 512 the call has not been concluded, the completed calls are discarded (step 530) after they get too old. Processing then continues again from step 502 by receiving the next incoming message.

If at step 504 the message is an SMDR message, the translation module at step 508 scans the list of recently completed calls. At step 510 the translation module calculates confidence factors for the recently completed calls by using the formula:

$$\text{Confidence Factor} = \sum_i ((\text{Match Quality})_i * (\text{Weighting Factor})_i)$$

If any matches are found (step 516), and more than one match is found (step 522), the match with the highest confidence factor is used (step 526). If only one match is found, that match is used (step 524). At step 540, the trunk channel information is extracted, and at step 544 the call record is updated within the list of recently completed calls. The call record is transmitted at step 548 to the rest of the recording system. At step 550 the call record is discarded from the list of recently completed calls. Completed call are discarded (step 530) after they get too old. If no matches were found at step 516, the completed calls are discarded (step 530) after they get too old. Processing then continues again from step 502 by receiving the next incoming message.

As shown in FIG. 6, the CTI Server can be viewed as a set of logically distinct layers that deal with translating and distributing CTI events. Starting from the bottom of the

US 6,728,345 B2

15

picture, CTI events flow from a PBX in its proprietary format to Dialogic CT-Connect middleware 640 another API layer 650 or custom interface layer 660 that each provide partial normalization of the data. This helps to reduce the complexity of the "translation" job, since there are fewer APIs than individual PBX types. But since one object of the subject system is to retain the flexibility to integrate with a variety of third-party CTI vendors (e.g., Dialogic, Genesys, etc.) there is another layer 670 above the API or custom interface layer to complete the job of "translation." The final result after passing through this "normalization" layer is that all of the CTI events are in a single, common, integrated data format.

Once the CTI events have been converted to a normalized format, the CTI Server can address its other mission of distributing (routing) the messages. The distribution layer 680 examines each message to determine what other recording system components need to receive it, and then sends a copy of the event to the appropriate destination(s).

This logical separation of responsibilities used in a preferred embodiment simplifies the programming required to implement the subject system. Translation modules do not need to know anything about other recording system components, and they can focus on dealing with a single specific PBX or vendor API layer. Likewise, the distribution module will not need to know anything about specific PBX or vendor API layers, and it can focus on making routing decisions and communicating with the rest of the recording system.

FIG. 7 illustrates how, in addition to telephony events, the CTI Server 710 used in accordance with the present invention is responsible for supplying certain metadata regarding agent events to the System Controller, which is part of the Scheduling & Control Services 130 shown in FIG. 1. This information, which generally includes agent ID, extension number, logon and logoff time, etc., is obtained when available from the various PBXs, ACDs, and turret systems. The agent events delivered to the System Controller 130 enable a map to be maintained of the extension number(s) where a real person can be found, at a given date and time. This information enables a browser application to intelligently associate some of the previously recorded calls even if a person was using different telephone sets according to a 'free seating' plan. The CTI Server 710 also keeps a local cache of the agent information, so that agent information can be included when sending the telephony events to the Call Record Generator 150.

The physical layout of the CTI Server used in a specific embodiment is shown in FIG. 8. With reference to FIG. 1, the translation modules are implemented by separate programs, such as CtiCtc.exe 406, which encapsulate the details on converting a specific PBX interface or vendor API layer into a normalized format. The distribution module is preferably implemented by a single program, CtiServ.exe 820, which includes the main processing and routing logic for the CTI Server.

As noted, the translation modules of the CTI Server convert proprietary-format CTI information into a normalized format. In accordance with a preferred embodiment, this is done in several layers within the program. The information is first converted by Dialogic's CT-Connect software into the CTC-API format, and then the conversion to the generic format used by the other components of the recording system is completed by the translation module CtiCtc.exe. Once the data is converted, it is transmitted to the distribution module (CtiServ.exe) by using a distributed communications method such as DCOM. Component

16

Object Model (COM) is a Microsoft specification that defines the interaction between objects in the Windows environment. DCOM (Distributed Component Object Model) is the network version of COM that allows objects running on different computers attached to a network to interact. An alternate embodiment of the CTI Server utilized Microsoft Message Queue (MSMQ) technology as the means to carry messages among the system components, instead of the original DCOM method used by CtiServ.exe, and those skilled in the art would appreciate that a variety of additional data communications technologies are also suitable to this role.

The translation module and the distribution module of the CTI Server can be located on different machines, if desired. There can be multiple translation modules running in the system—one for each PBX or CTI middleware environment. There can also be different types of translation modules, with one version for each interface or API layer. As depicted in FIG. 8, CtiCtc.exe deals with the Dialogic CT-Connect API, and there are 3 copies of this program running to handle the PBXs. If other types of APIs are used, there would be other programs for these various interfaces. All translation modules contribute data upward to the distribution module in a single, common, normalized format. An example of a version of CtiCtc.exe configured to work with a Lucent Telephony Services interface (and thus called CtiLts.exe instead of CtiCtc.exe) is shown in FIG. 9. The modules which are common to both versions of the program are shown in FIG. 9 as shaded gray. The unshaded modules represent those portions of the program that necessarily vary between CtiCtc.exe and CtiLts.exe, due to the differing input parameters and data structures used by both systems.

Again with reference to FIG. 8, the distribution module (CtiServ.exe) receives and collects all the CTI events from the various translation modules. Then it puts the events into a single inbound queue 830 for processing by a main control thread 835. After the events are processed, they are separated into individual outbound queues 840. Finally, the events are sent by various delivery threads 850 to the CRG components within different Voice Servers. The main processing thread 855 (WinMain) is deliberately isolated (decoupled) from the inputs and outputs to ensure that delays in transmitting or receiving data will not impact the overall performance of the CTI Server.

FIG. 11 shows how the CTI server in accordance with a specific embodiment consists of several threads that implement three distinct layers of processing (data collection 1110, data normalization 1120, and message emission 1130). FIG. 12 illustrates the processing steps of these layers. The dashed lines indicate message flow between threads, whereas the solid lines indicate program logic flow. The CTI translation modules are thus internally separated into 3 major sub-tasks; (1) data collection from the input source (PBX, CTI middleware, etc.); (2) normalization of the data to a common format; and (3) communications with the system platform.

In a data collection layer, the initial step 1210 is to open the connection to the CTI data source. At step 1214 the layer receives a CTI event, and at step 1216 posts the CTI event to the Message Queue 462 (see FIG. 4). If at step 1218 a shutdown is in progress, the connection to the CTI data source is closed at step 1220, and at step 1222 data collection is ended. If at step 1218 a shutdown is not in progress, the CTI connection remains open (step 1212).

At step 1228, the data normalization layer receives a CTI event from the Message Queue 462. The data normalization layer updates the telephony model at step 1230. See FIG. 13

US 6,728,345 B2

17

for a more detailed explanation of the updating of the telephony model. At step 1231, the call state is posted to the Message Queue, if necessary. At step 1232 completed calls are discarded from memory after they age beyond a configurable time limit. At step 1233 the "hang-up" routine is called to update the telephony model for held or bumped calls after they age beyond a configurable time limit. At step 1234, if a shutdown is in progress, the data normalization layer checks the inbound message queue at step 1236. If the message queue is empty, data normalization is ended (step 1238). If the message queue is not empty at step 1236 or if there is not a shutdown in progress at step 1234, the data normalization layer goes to step 1226 and waits for the next CTI event to arrive.

The message emission process begins with opening a connection to a target platform, such as the da Vinci or SymphonyCTI recording systems at step 1240. At step 1244, the message emission layer receives the call state from the message queue 462. At step 1246, the call state data is converted into a platform-specific format. At step 1248, the message emitter sends the message to the target platform. At step 1250, if a shutdown is in progress, a check is made at step 1252 for whether the inbound message queue is empty. If the inbound message queue is empty, message emission is ended at step 1254. If the inbound message queue is not empty at step 1252, or if there is not a shutdown in progress at step 1250, the message emission layer, at step 1242, maintains the open connection to the target platform and awaits the next call state transmission.

Master Call Record

The CTI Server sends "Call Event Records" onward to the recording platform. These messages provide details on the start and end of calls, as well as significant transitions that affect the lists of participants for the calls. The list of participants is cumulative, and information regarding participants is retained for the entire duration of the call even when some participants in the list may have dropped off from the call. If a participant rejoins the call, a new, separate entry will be created to reflect that change within the participant list. The following table shows the fields contained within these messages.

CtiCallEvent		
Name	Type (max length)	Description
Version	WORD	Version number of this message format, for reverse compatibility.
MessageID	GUID	Unique ID for this message instance
RecorderNode	WORD	A number that identifies a particular Voice Server
RecorderChannel	WORD	A number that identifies a recording input channel on a Voice Server
EventType	BYTE	Indicates if this event added (0x01) and/or dropped (0x02) participants in the call.
EventReason	BYTE	Indicates if this call was affected by a normal (1), conference (2), or transfer (3) telephony event
CTICallRecId	GUID	Unique ID pertaining to entire call (CTI server provides the same ID for a call that is transferred, conferenced, etc)
CallIDirection	BYTE	Indicates call origin - outbound (0x12), inbound (0x21), internal (0x11), or unknown (0x44)

18

-continued

CtiCallEvent		
Name	Type (max length)	Description
RingLength	WORD	Seconds between the first ring signal and going off-hook (picking up the phone)
DTMFCode	String*(50)	DTMF codes entered during the call
ApplicationData	String*(32)	Character array dedicated to information the switch may provide along with the call (e.g., account number)
CallingParty	WORD	Index number of the calling party within the participant list. Normally this is zero.
CalledParty	WORD	Index number of the called party within the participant list. Normally this is one.
PBXCallRecId	DWORD	Number provided by the PBX to identify this call.
NumberOfParticipants	WORD	Count of participants in the following array.
ParticipantList	Vector*	Array of PartyListElement describing all participants involved in the call

*ObjectSpace data types

ObjectSpace is a set of C++ class libraries provided by ObjectSpace, Inc., that define useful general-purpose data structures including representations of strings, time values, and collections of objects (such as vector arrays or linked lists). These class libraries are implemented in a way that supports a wide variety of computer operating systems. Those skilled in the art will appreciate that many alternate implementations for such data structures are suitable for this role.

CtiPartyListElement		
Name	Type	Description
AgentID	String*(24)	Registered ID of person, typically used for "free seating" call center environments
Number	String*(24)	Telephone number of this participant (e.g., ANI, DNIS, Dialed Digits)
Consol	String*(10)	Seating position that can consist of one or more stations.
Station	String*(10)	Unique telephone set, possibly with multiple extensions.
Extension	String*(6)	Internal line number of the participant
SwitchId	WORD	Number of the switch (PBX, ACD, or turrel system) which is handling the conversation
TrunkID	WORD	Identification of trunk line which is handling the conversation
VirtChannel	WORD	Identification of trunk's channel (time slot) which is handling the conversation
Location-Reference	BYTE	Describes the location of participant with respect to the switch - can be internal (1), external (2), or unknown (3)
StartTime	time_and_date*	Time participant joined the call
EndTime	time_and_date*	Time participant left the call
ConnectReason	BYTE	How participant joined the call: norm start of call (1), being added to a conference (2), or receiving a transferred call (3)

US 6,728,345 B2

19

-continued

<u>CTIPartyListElement</u>		
Name	Type	Description
Disconnect-Reason	BYTE	How participant left the call: normal end of the call by hanging up (1), dropping out of a conference (2), transferring away a call (3), or call ends by another party hanging up (4).
Changed	BOOL	Indicates if recent change in CTI message:

*ObjectSpace data types

For external participants, only the fields Number, SwitchName, TrunkID, VirtChannel, LocationReference, StartTime, EndTime, ConnectReason, and DisconnectReason will be applicable. For internal participants, all fields may be applicable. Unused string fields will be null terminated. Unused number fields are set to zero. Each call event record will contain at least two participants in the list. These two participants are the original calling party (0) and called party (1) and will appear within the list in that order respectively.

Note: The data field "Number" will be filled in a variety of ways, depending upon the type of participant and direction of the call.

Participant Type	Call Direction	Number Field
External participant	Inbound Call	ANI
External participant	Outbound Call	Dialed Digits
Internal participant	Inbound Call	DNIS or Extension
Internal participant	Outbound Call	Extension
Internal participant	Internal Call	Dialed Digits or Extension

The CTI Server sends "Agent Event Records" onward to the recording platform's System Controller to convey information when an agent logs on/off at a particular location. The following table shows the fields contained within these messages.

<u>CTIAgentEvent</u>		
Name	Type (max length)	Description
Version	WORD	Version number of this message format, for reverse compatibility.
MessageID	GUID	Unique ID for this message instance
EventType	BYTE	Indicates if this event pertains to either a logon (1) or logoff (2).
LocationType	BYTE	Indicates if this event pertains to a location type such as a console (1), station (2) or extension (3).
AgentID	String*(24)	Registered ID of person, typically used for "free seating" call center environments
SwitchID	WORD	Number of the switch (PBX, ACD, or turret system) where the agent connected.
Console	String*(10)	Seating position that can consist of one or more stations.
Station	String*(10)	Unique telephone set, possibly with multiple extensions.

20

-continued

<u>CTIAgentEvent</u>		
Name	Type (max length)	Description
Extension	String*(6)	Internal line number of the participant
StartTime	time_and_date*	Time that the agent logged in.
EndTime	time_and_date*	Time that the agent logged out.

*ObjectSpace data types

Within any given "Agent Event Record", only one of the following three fields will be applicable: Console, Station, or Extension. The actual mapping is determined by the LocationType. Unused string fields will be null terminated. Unused number fields are set to zero.

It will be appreciated that the general principles behind the method described above are suitable not only for associating and combining real-time CTI data with the trunk channel information from an SMDR message, but also for any situation where a mixture of information is being provided from two or more sources and there is a need to gather and merge the information to get a more complete picture of what is actually happening in the system. The disclosed method could easily be adapted by those of ordinary skill in the art to situations in which the mapping or association between the multiple sources of information is "weak" and prone to ambiguity. While this method does not make the potential ambiguity disappear, it helps to define a quantitative set of rules for making a judgement call on when a match is "good enough" to act upon. While human beings are often capable of making such judgement calls intuitively, computers need a specific set of instructions in order to act in a repeatable and reliable fashion upon the input data.

Previous recording systems that made use of CTI to collect enhanced search information mimicked the event-oriented interfaces provided on the data links from a PBX. Individual database records were constructed on a 1-to-1 basis for the events occurring during the total lifetime of a phone call. The interpretation of the series of events was left to the end user. Associations between related events were made difficult in certain cases because the call identification numbers given by a PBX may change after a call has been transferred or conferenced, or the numbers may be recycled and reused over time. Following and tracing the history of events for a complete call from the perspective of the external customer could require much manual and repetitive searching. Playing back the entire set of audio recordings from the start of that customer's interaction with the business, to the ultimate conclusion of that customer's transaction, could also require additional repetitive manual requests to play back the individual recorded segments within a call that was transferred or conferenced.

To resolve this problem, the CTI server of the preferred embodiment maintains and accumulates information within a data model of telephony activity. FIG. 10 depicts the key elements of the data model. This consolidated information is shared with the rest of the recording system when parties join or leave a call, thereby eliminating the need for downstream components to store or interpret the individual CTI events occurring during a call's lifetime.

During the active lifetime of a call, real-time information is accumulated within a historical call record that tracks each participant within the call. At certain key points during the accumulation of data, whenever a party joins or leaves the

US 6,728,345 B2

21

conversation, the call record is transmitted onward to allow the rest of the recording system to process the information accumulated to that point. Upon the conclusion of the call, the CTI server of the preferred embodiment retains a copy of the call record for a configurable time interval before discarding it from memory. This delay allows for the arrival of the SMDR data.

The call records are organized into a two-tiered hierarchy of calls and participants. Certain data fields that apply globally to the entire call are stored at the upper level. Most data fields, however, apply only to a specific party involved within a call, and are stored at the lower level. Individual participants can have identifying information (such as extension number, agent ID, telephone number via DNIS/ANI/CLID, trunk and channel) along with time-stamps and reason codes for the entry and exit from participation in the telephone call. Reason codes include initial start, transfer, hold, resume, conference add/drop, and hang-up.

The currently active call on each telephone set being monitored is maintained within a storage area 1020 of the data model. Also, the data model provides for an open-ended list 1040 of calls that may be "on hold" (and therefore not associated with any telephone set). There is also a list 1030 that can be used temporarily for calls when they are in a state of transition during transfers, queuing or re-routing, for the brief period of time when an active call is disassociated from its original telephone set but not yet associated with a new telephone set. Finally, there is a list 1050 of recently completed calls that is used to await additional information that might be provided from a SMDR message.

This complete set of data structures is replicated independently for each CTI server that monitors a separate PBX within the overall call center environment.

The call-centric structure and the list of participants facilitate a common framework for modeling the various types of complex call scenarios that may occur during the life of a call, far beyond the simplest example of a basic two-party telephone call. Moreover, the recording units can link references (i.e., logical pointers) to the audio recordings for a portion of the call, so that these audio sections are associated with the total history of the logical telephone call. Each call record can be linked within the database to an open-ended list of references, which provides: the name of a Voice Server; the name of a .WAV file containing the audio recording; the offset within the .WAV file to the start of the recording segment; the start time of the recording segment; and the duration of the recording segment.

Rather than relying exclusively upon the call identification number assigned by the PBX, the CTI server of the preferred embodiment obtains a Globally Unique Identifier (GUID), that is generated at the software's request by the underlying Microsoft Windows NT operating system, and uses that GUID to identify the call uniquely within the recording system's memory, online storage database, and offline storage archives. The GUID is initially requested at the start of the call. While the call remains active, the CTI server maintains a record of both the call identification number assigned by the PBX, and the GUID assigned to the call by the software of the preferred embodiment. When a CTI event arrives, the system searches the telephony model to find a matching call record for the PBX-assigned call identification number. At transition points during a call's lifetime, such as when it is transferred or conferenced, the PBX typically provides the old and new identification numbers together in that single transition event. In these cases, after locating the matching call record, the software of the preferred embodiment updates its record of the call identi-

22

fication number now being used by the PBX while retaining the originally allocated GUID value. In this way, the same GUID identifies the call throughout its lifetime, even while the PBX call identifier may be changing. The long-term uniqueness of the GUID value is also useful if the PBX recycles and reuses previously assigned call identifiers. It further helps in dealing with calls within a multiple PBX environment. While another PBX may coincidentally use the same call identification number, a different GUID is assigned at the start of each individual call, thereby avoiding a conflict within the telephony model.

As shown in FIG. 11, the CTI server consists of three distinct layers. Each layer actually runs in a separate thread of execution, and communicates with the other layers through shared memory, control semaphores, and message queues. The first layer 1110 is responsible for gathering input from the PBX data link(s), and there can actually be several threads running to provide better throughput capacity or to handle multiple diverse input sources (e.g., SMDR and real-time CTI messages). After saving the clock time when a message is received, the first layer 1110 places the message into a queue for subsequent processing by the second "analyzer" layer. The second layer 1120 is responsible for updating and maintaining the telephony model within the memory of the CTI server, and for deciding when to send copies of call records onward to the rest of the recording system. When a call record needs to be sent onward, the call record is placed into a message queue for subsequent processing by a third "message emitter" layer 1130, which is responsible for communications with other components of the overall recording system. This separation of layers gives the CTI server the flexibility to process its input and output sources in a de-coupled fashion, so that any delay in one area of communications does not affect the processing of another area. In a sense, the design approach provides a virtual "shock absorber" so that bursts of input traffic, or temporary lag times in communicating with other parts of the recording system, can be tolerated without loss of data or incorrect operation of the system.

The call records saved within the telephony model also include a record of the last state of the device as reported by the PBX. This information is used by the analyzer to run state machine rules, in order to select a handler routine for a subsequent message. The CTI server uses the previous state of the device (e.g., ringing, answered, and so forth) along with the current state of the device to select a handler routine from a matrix of potential choices.

The analyzer layer is of particular interest, since it is responsible for updating and maintaining the data model of telephony activity. Its overall program logic flow is illustrated in FIG. 12 and the subroutine called at step 1230 is shown in further detail by FIG. 13. This program logic is described below.

1. Receive a CTI event from the message queue at step 1228.
2. Enter the subroutine at step 1230 to update the telephony model. Referring now to FIG. 13, search the data of model of telephony activity, to find a matching record at step 1322 with the same monitored device (i.e., telephone set).
3. If the PBX-assigned call identification number does not agree, search for a matching record in the lists of calls on hold, in transition states, or recently completed. If a match is then found, move the call record on the affected device to the list of calls in transition states, and move the matching record to the monitored device.
4. At step 1324, use the previous state as recording within the telephony model, along with the new state reported in the CTI event, to select the appropriate handler routine at

US 6,728,345 B2

23

step 1332 from a matrix of choices. The handler routine will be one such as those described below.

5. At step 1340, run the steps of the handler routine. This will commonly include steps to save at step 1342 information from the CTI event into the call record, to update the call-related portion of the Object Status, if necessary (step 1344), to update Participants within the Object Status, if necessary (step 1352), to run additional action methods or handler routines for other affected telephony objects, if necessary (step 1348), and to post Object Status to the message Queue for the Emitter to a target platform (step 1354).
6. At step 1360, returning to FIG. 12, at step 1232, discard completed calls within the data model of telephony activity, if they have aged beyond a certain re-configurable time limit.
7. Call the "hang-up" routine at step 1233 for any held call that have aged beyond a separate re-configurable time limit. Likewise, call the "hang-up" routine for any calls marked in transition, which have aged beyond another separate re-configurable time limit.
8. Continue again from the beginning of this logical program flow at step 1226.

The following description lists processing steps for various handler routines that may be called in response to certain event types using a decision matrix based upon past and current state information.

Handler Routines	
Ignore:	adjust state based on CTI event
DialTone:	save the initial start-time of the call save the original dialed number, if available adjust state based on CTI event
RingIn:	adjust state based on CTI event event time-stamp when ring occurred clear call record set inbound, outbound, internal
Answer:	adjust state based on CTI event compute total ringing duration fill in call record with calling party & called party generate START message to recording system
Abort:	adjust state based on CTI event clear timers & original dialed number
Hang-Up:	adjust state based on CTI event update call record to stop all parties indicate which party actually hung up on the call generate STOP message to recording system
RingOut:	adjust state based on CTI event time-stamp when ring occurred (i.e., now) clear call record set inbound, outbound, internal compute total ringing duration (i.e., zero) fill in call record with calling party & called party generate START message to recording system
Hold:	adjust state based on CTI event stop participant placing the call on hold add new placeholder participant for HOLD generate TRANSFER message to recording system move call record to hold area fill device slot with a new empty call record
Resume:	if device slot not idle, move call record to transition list move matching call record from hold area to device slot adjust state to "active" stop the placeholder participant for HOLD add new participant for telephone set that resumes the call generate TRANSFER message to recording system

24

-continued

Handler Routines	
Conference:	if call record found in hold area, if device slot not idle, move call record to transition list move matching call record from hold area to device slot adjust state to "active" stop the placeholder participant for HOLD add new participant for telephone set that resumes the call generate TRANSFER message to recording system adjust state based on CTI event add new participant for telephone set that is added via conference generate CONFERENCE-ADD message to recording system
Transfer:	if call record found in hold area, if device slot not idle, move call record to transition list move matching call record from hold area to device slot adjust state based on CTI event stop the participant leaving the scope of the call (either a device or HOLD) add new participant receiving the transferred call generate TRANSFER message to recording system adjust state based on CTI event
ConfDrop:	stop the participant leaving the scope of the call generate CONFERENCE-DROP message to recording system
OpAnswer:	adjust state based on CTI event re-compute total ringing duration correct the affected participant entry in the call record generate CORRECTED message
DestChanged:	clear call record the call will be processed via a subsequent CTI event

The following step-by-step description describes the same call scenario as in FIG. 3, but with emphasis on the data model of telephony activity.

1. A real-time CTI message occurs describing that phone B is ringing, but not yet answered.
2. The "Ringin" routine is invoked.
3. The telephony model is updated with the time when ringing started (for use later in measuring ring duration) and the call direction. These facts are stored with device B 340.
4. A real-time CTI message occurs describing the start of the call between A 335 and B 340.
5. The "Answer" routine is invoked.
6. The telephony model is updated to reflect the initial 2 participants (A and B) started normally at t0 310.
7. A copy of the call record is sent onward to the rest of the recording system.
8. The call record is retained within the telephony model, associated with device B 340.
9. A real time CTI message occurs describing that B 340 placed the call on hold.
10. The "Hold" routine is invoked.
11. The telephony model is updated to reflect that B 340 transferred the call to HOLD 345 at t1 315. (This information is accumulated with the information previously gathered at t0).
12. A copy of the call record is sent onward to the rest of the recording system.
13. The call record is removed from device B 340 within the telephony model, but kept in a list of held calls.
14. A real-time CTI message occurs describing that B 350 returned to the call and invited C 355 by conferencing.
15. The "Conference" routine is invoked.
16. The call record is moved within the telephony model from the list of held calls back to device B 350.
17. The telephony model is updated to reflect that HOLD 345 transferred the call back to B 350 at t2 320. (Note that information is accumulated with the information previously gathered at t0 and t1).

US 6,728,345 B2

25

18. A copy of the call record is sent onward to the rest of the recording system.
19. The telephony model is updated to reflect that C 355 joined the call as a conference participant at t2 320. (This information continues to be accumulated with previously gathered information).
20. A copy of the call record is sent onward to the rest of the recording system.
21. The call record is retained with both devices B 350 and C 355 within the telephony model.
22. A real-time CTI message occurs describing that C 355 dropped out of the call.
23. The "ConfDrop" routine 386 is invoked.
24. The telephony model is updated to reflect that C dropped out of the conference at t3. (This information continues to be accumulated with previously gathered information).
25. A copy of the call record is sent onward to the rest of the recording system.
26. The call record is removed from device C within the telephony model, but retained with device B.
27. A real-time CTI message occurs describing that A terminated the call.
28. The "Hang-Up" routine is invoked.
29. The telephony model is updated to reflect that A stopped normally and B stopped because the other party hung up at t4 330. (This information continues to be accumulated with previously gathered information).
30. A copy of the call record is sent onward to the rest of the recording system.
31. The call record is removed from device B 350, but kept in a list of completed calls.
32. A SMDR message occurs summarizing the call in its entirety.
33. The list of completed calls is searched to find a match, and the appropriate call record is retrieved.
34. The call record is updated with the trunk channel information from the SMDR message.
35. A copy of the call record is sent onward to the rest of the recording system.
36. The call record is removed from the list of completed calls.

FIG. 14 depicts the flow of information within the remainder of the recording system. The same enhanced search information S1 1412 is provided by the CTI server to all of the recording units involved in handling a portion of the call. Even if a call is transferred to another telephone set, which is attached to an input channel on a different recorder, the entire call will still remain associated as one entity within the system. Each recorder maintains a local copy of the audio sections V1 1416, V2 1420, and V3 1424 that it obtained during the call, along with a complete call record containing search information S1 1412 which contains the two-tiered call and participant model. The search information is copied to a central database server 1450, along with references (i.e., logical pointers) to the original audio recordings VR1 1428, VR2 1432, and VR3 1436. When a user searches for a call, the search results 1465 will include the complete call record S1 1412. By using the audio references the playback software can reassemble the complete audio for the original call, including sections possibly obtained from different physical recording units.

The general principles behind the method described above would be suitable, not only for representing the complete history of telephone call's lifetime, but other forms of multi-party communications. This may include certain forms of radio traffic that have an associated data link, which provides "talk group" identification numbers (or similar types of descriptive search data in relation to the audio traffic).

26

Call Recorder Generator

The Call Record Generator (CRG) in accordance with the present invention performs the function of combining voice and data into call records. It performs this function at or near real time. The CRG, when combined with the metadata normalization module CTI Server, makes up a system that can be used in current and future communication recording products.

The CRG is responsible for collecting data from different sources with respect to portions of a call on various recording input channels, and merging them together into a unified call record. One of these sources is the recorder that creates the files containing media. Another source provides metadata describing the when, who, why and where information of a call. This call record metadata comprises the start and stop times of a segment within a call, as well as CTI data such as telephone numbers and agent IDs. These metadata sources include but are not limited to Telephony switches and Trunked Radio servers. The CRG depends upon the CTI Server to normalize data from these sources.

FIG. 1 illustrates the relationship between the CRG and the rest of the system. Since call records are an essential part of the recording system, there is one CRG dedicated to each recorder and physically located in the same Voice Server. If other system components become inoperable, call record generation will remain functional (albeit at a reduced level).

The CTI server supplies switch events to the appropriate recorder indicating either the status of calls or providing data for population. The CTI server provides, along with call record data, the association between the recorder location (i.e., Voice Server and recording input channel number) and the switch connection point. The switch connection point is described as either the extension for extension side recording or the Trunk ID/virtual channel (TDM time slot) for trunk side recording. In addition to this mapping, an agent identification will be supplied for agents currently associated with this call. The recorder location, switch identification and corresponding agent are stored in the call record. The CRG is designed to work with many different configurations of the disclosed system. These configurations include: systems without CTI Servers; systems with Real-time CTI Servers; systems with non-Real-time CTI Servers; recorders with analog inputs; recorders with digital inputs; recording on the trunk side of the telephony switch; and any combination of CTI Servers, Recorder inputs, and recorder positions mentioned above.

Due to the non-standard operation of telephony switches and flexibility requirements of the recording device, the CRG must handle event data arriving in different chronological order. In accordance with a preferred embodiment, it accomplishes this by requiring all events to indicate time of occurrence and maintaining a history of them. A call record can be created solely from either event sources but when both are present, call records are generated using recorder information together with CTI data.

It is clear that the use of different data sources and non-synchronous messages, as required to support various alternative configurations of the overall system, add considerable complexity to the CRG. For example, with the many different objects supplying information for a particular call, the messages from each can be received in any order. The CRG must be able to accommodate this requirement. In some configurations, objects supply redundant information to the CRG. The CRG provides a mechanism for selecting which information will populate the call record.

In the most basic mode of operation, the CRG has no CTI input and is recording solely on VOX events from the

US 6,728,345 B2

27

recorder controller (the term "recorder controller" is used interchangeably herein with "Audio Recorder"; both terms refer to the software that primarily directs the processing of the audio data). VOX is Dialogic Corporation's digital encoding format for audio samples. This term is also sometimes used to refer voice-activated initiation of recording, a process that conserves storage space since a continuous recording process would include periods of silence. These VOX events mark the beginning of energy activity on a phone line and are terminated by the lack of activity. With this approach, an actual phone call may include several call records. To address this problem, the recorder waits a configurable holdover period while silence is present before terminating an active VOX clip (the term "Recorder" is used interchangeably herein with the term "Voice Server"; both terms refer to the physical recording server). The goal is to concatenate parts of a phone call where gaps of silence exist. The solution lies in determining an appropriate holdover time so as to avoid merging audio from the next phone call if it occurs close to the end of the last call.

The next level of operation is where the recorder hardware can detect telephony signaling such as off hook and on hook. The CRG has no CTI input from the switch and is recording solely on events from the recorder controller, but these events mark the beginning and end of a phone call (off hook and on hook). The resultant call record reflects a phone call in entirety but lacks much descriptive data that accompanies switch data.

The highest level of operation involves the use of a CTI Server. In this configuration, the CRG receives recorder events as well as CTI events. Since CTI events give the CRG a description of the entire phone call, information obtained from them drive the creation of call records. Recorder data describing audio events are absorbed into the CTI call record whenever audio and CTI times overlap. With CTI events driving call record generation, non-audio based call records can be created.

Mixing of recorder and CTI data occurs by comparing ranges of time indicated. For example, a person whose telephone extension is being recorded is involved in a phone call for a given period of time. The recorder events indicating that audio was recording on the same extension during the same time period are associated with the CTI metadata for that phone call. Since the data from the CTI Server may arrive before or after the corresponding recorder events, the CRG maintains an independent history for each type of data.

For the case where CTI events arrive before the recorder events, the CTI events are added to the CTI history list. When the corresponding recorder events arrive, the CTI history list is swept for matching time ranges and associations are made when they occur. For the case where recorder events arrive before the CTI events, the recorder events are added to the recorder history list. When the corresponding CTI events arrive, the recorder history list is swept for matching time ranges and associations are made when they occur.

Previous recording systems stored voice data and metadata in separate locations. A significant disadvantage to this approach is that it is left up to the other software subsystems to combine the information when required. This approach makes the work of other system features, such as playback and archiving to offline storage, more complicated and prone to error. By performing this "early binding" of the audio and CTI data in accordance with the present invention, such problems are avoided and the above desirable features are therefore much simpler to implement in a correct, robust fashion.

28

When attempting to playback media for a given call record, the playback mechanism must figure out where the audio for the call record exists and when determined, retrieve and locate the start time inside this media. The CRG places this media metadata in related tables, thus informing the playback mechanism what files are associated, their location, and what time ranges inside the file are available for playback.

Most communication systems require an archive mechanism to store large amounts of data that cannot be kept online due to capacity limitations. The CRG used in accordance with this invention assists with archiving by allowing both call record metadata and the media files to be stored on the same offline media. Current versions of recording systems store call record metadata and media files on separate offline media making restore operations more complicated.

For enhanced security purposes in a preferred embodiment, the CRG accesses media files associated with a call record through the use of media segmentation. A media segment includes, in addition to a media filename and location, a start time and duration inside the media file. Media segmentation is necessary when creating CTI based call records since a call record may involve many recording locations throughout the life of the call. The specified time range isolates a portion of the media file that can be accessed through this call record. This feature is very important when there are many call records located in one media file. A user attempting to play back media of a call record, to which he has the permission for access, may or may not have permission to play back other call records sharing the same physical file.

The Call Record Generator is responsible for merging CTI search data and a multitude of voice recording segments together into a single manageable unit of data. This software includes a flexible receiver algorithm to allow voice and search data to arrive in either order, without requiring one to precede the other. Once combined, the call record can be managed as a single entity, which greatly simplifies and reduces the work necessary to perform search, retrieval, and archival operations. This approach also offers a more natural and flexible framework for controlling security access to the recordings, on an individual call basis (or even on selected portions within a call).

As shown in FIG. 15, a recording unit operating with only voice signaling to guide the creation of its call records could make a number of fragmented audio segments. When the recording unit is supplied with CTI search data giving a complete history of the call's lifetime, and when it is designed to merge the CTI search data and audio segments into a combined unit of Voicedata™, the results can simplify and reduce the work necessary for a user to obtain a desired call from the system. Several audio segments can be grouped together, and can be understood by the system as being part of the same logical telephone call. It is also possible that a single audio segment was recorded, even though parts belong to separate telephone calls, because the delay between stopping the first call and starting the second call was very brief. Without a sufficient silence gap, it may appear to the voice recording unit that this was a continuous segment of audio, rather than belonging to two separate calls. When the CTI search data is merged with the audio segments, the system can use this information to recognize when an audio segment should be split and divided between two logically distinct calls.

The purpose of the Call Record Generator (CRG) is to collect information describing multimedia data and store it

US 6,728,345 B2

29

in a central location. The CRG produces Master Call Records (MCRs) that encapsulate information describing a phone call as well as the location multimedia that is associated with it. This description data comes from a multitude of sources including but not limited to a Voice Server and CTI Server. Likewise, the design of the system envisions that there will be a number of possible input sources for audio recording.

Whatever the means for collecting CTI information, it is communicated to the rest of the system in a common, normalized format. The CTI information is passed from the translation modules to a message router. From that point, copies of the information are sent to the scheduling and control services and to the CRG for the appropriate recorder (s). The scheduling and control services are responsible for starting and stopping the audio recorder, according to pre-defined rules that are dependent upon time and CTI information. The CRG is responsible for merging the audio recording with the CTI information to determine the temporal boundaries of the call and prepare the Voicedata for storage.

The user workstation typically searches and retrieves records from the Voicedata storage, and then obtains audio for playback directly from each recorder's private storage area. The user workstation can also be used to monitor "live" conversations by communicating directly with the recorder. The user workstation can also control the audio recorder indirectly by manipulating the rules used by the scheduling and control services.

In the preferred embodiment, the user workstation has software that is configured to display a graphical user interface (GUI) such as that shown in FIG. 16. The GUI in FIG. 16 uses the information compiled in the Master Call Record to generate a graphical representation 1610 of the call, as well as displaying the call record information in alphanumeric form in a table 1620. Further, when the call is played back, the displayed segments in the graphical representation are highlighted to indicate the portion of the call being played back. For example, in FIG. 16, if the entire call is played back, when the portion of the call that occurred between 6:20:08 AM and 6:55:31 AM is played back the bars 1632, 1634, and 1636 are highlighted from left to right as the call is played back. Thus, as the part of the call that occurred at 6:55:31 AM is reached, bar 1634 is fully highlighted, and bars 1632 and 1636 are highlighted starting from the left and extending to those points on bars 1632 and 1636 that are directly above and below the right-hand endpoint of bar 1634. After the played back call reaches the part that occurred at 6:55:31 AM, the bar 1638 begins to be highlighted starting at the left endpoint. When the part of the call that occurred at 7:10:22 AM is reached, the bar 1636 is fully highlighted. At that point, the bars 1632 and 1636 are highlighted from their left-hand endpoints and extending to points directly above the right-hand endpoint of bar 1638. The process continues as long as the call is being played back, until bars 1632, 1634, 1636, 1638, 1642, and 1644 are completely highlighted.

In alternate embodiments of the subject invention, playback of a portion of a call can be activated directly from the graphical view by mouse-clicking or by selecting from a pop-up menu; circular "pie-charts" show the percentage of time for each party involved during the lifetime of the call; an animated vertical line scrolls along to indicate the progression of time when the call whose graph is being displayed is played back; and miniature pictorial icons are shown within the graphs to indicate start stop reasons, type of participant, etc. All of these embodiments are enabled by the data contained in the Master Call Record.

30

As a method of managing complexity, the preferred embodiment of the system uses data abstraction to isolate the internal details of certain structures to those components which need to operate directly upon them. Information is organized by the collectors (or producers) of that data, into a digested form that is more easily usable by the applications which need to retrieve and process the data.

For example, the CTI translation modules supply normalized records to the rest of the system in a common shared format, rather than exposing the details of various different CTI links. The system data model is call-centric, containing a detailed cumulative ("cradle to grave") history, rather than event-centric, which would place the burden of work on the receiving applications. Likewise, agent information is session-oriented rather than event-oriented.

Whether collecting information from a CTI link, or recording audio from a telephone call, a fundamental design advantage for the system of the preferred embodiment that it operates virtually invisibly, from the end-user's perspective. The system architecture is designed to avoid any interference with the normal operation of a call center environment.

For example, the CTI translation modules are focused exclusively on collecting and normalizing information that is to be supplied to the rest of the system. Liability recording systems, and quality monitoring systems that use "service observance" techniques, do not require any active call control on the CTI links. Only the technique known as "dynamic channel allocation" requires active call control through CTI links to establish a "conference" or "bridge" session between the audio recorder and the telephone call participants. When active control is required to implement such a feature, it can be implemented through a new logically separate task, without significantly affecting the rest of the system design. For customers that have existing CTI infrastructure and applications, the system will not interfere with their existing operations.

The CRG is responsible for collecting data from the CTI Server, creating CTI-based call records, and attempting to match those records with existing recorded audio data. If the CRG receives CTI information indicating that audio data for the same call resides on two or more recorders (for example, due to a transfer), records will be generated for each portion with a common call record ID. This ID can later be used to query for all of the pieces ("segments") comprising the complete call. Each segment will identify the recorder that contains that piece of the call.

During playback, a player module connects to a program located on a Voice Server called the Playback server ("PBServer"). The machine name of the particular Voice Server which holds an audio segment is stored by the CRG in the call record table within the Voicedata storage, and is passed into the player module after being extracted by a User Workstation's sub-component known as the call record browser. A call record playback request is then submitted, which causes the PBServer to query for a specific call record's audio files located on that physical machine, open them, and prepare to stream the audio upon buffer requests back to the client software (the player module) on the User Workstation. If successful, a series of requests is then issued from the client, each of which will obtain just enough audio to play to a waveOut device while maintaining a safety net of extra audio in case of network delays. Upon a request to "move" within the scope of a call record, the PBServer repositions its lead pointer to the desired location and then begins passing buffers from that point. This series of

US 6,728,345 B2

31

Request and Move commands continues until the user chooses to end the session by shutting down the client-side audio player.

As used herein, the term "Call Control" refers to the part of the metadata concerning the creation and termination of call records. The term "Media" refers to the actual data that is being recorded. This term is used interchangeably with audio since the primary design of the CRG is to support audio recording. However, the CRG could apply to any data being recorded including multimedia or screen image data. The term "Metadata" refers to informational data associated with multimedia data describing its contents. The term "Call Participant" refers to an entity that is involved in a phone call. There are at least two participants involved in a call; namely the calling and called parties. Participants can consist of people, VRUs, or placeholders for parties being placed on hold. The term "Recorder Participant" refers to a participant in the MCRs Participant list who is located at the same connection point on the Switch to which the recorder input channel is connected. In accordance with the present invention, there can be more than one Recorder Participant associated with a call record since participants can enter and leave many times in a call. For any given recorder channel, there can only be one matching Recorder Participant active (not disconnected) at any given time across all call records associated with that channel. A "VOX-based Master Call Record contains information contributed by events from the Recorder alone, in the absence of data from a CTI Server. A VRU is a Voice Response Unit: an automated system that prompts calling parties for information and forwards them to the appropriate handler.

Once a recorder channel becomes involved in a phone call, it will be associated with all subsequent CTI events pertaining to the same call. This occurs even if the recorder location is no longer involved in the call. As an example, consider a phone call involving a transfer. FIG. 16A shows the subject system containing a CTI Server 710 and Recorder 1640. A recorder channel 0 1650 is attached to the extension side to extension 0001 1622. A phone call is initiated from the outside by some agent "A" 1602 and initially connects to agent "B" 1608 at extension 0001 1622. Agent "B" 1608 places "A" 1602 on hold and transfers him to Agent "C" 1612 at extension 0002 1630. The CRG recording extension 0001 1622 would receive all update messages with regard to this call since he/she participated in the call. Descriptive information from the CTI Server 710 would look like that in table 1600 in FIG. 16B. Audio clips recorded while agent "B" 1608 was involved in the call are recorded in a VOX based call record as shown in FIG. 17. The three media files created from the conversation may overlap with the recorder participant (agent "B"). At some point, determined by the order by which recorder and CTI events are received, audio data information from the VOX call record is absorbed into the CTI MCR for the times the recorder participant is involved (see the results after the sweep of the VOX and CTI history lists). For this call record, audio recorded between times t_1 and t_4 is absorbed. Any remaining audio is left in the VOX MCR for possible absorption in other CTI MCRs adjacent in time to this one. Since extension 0001 in this call record is different from the other participants in that it is associated with the same switch point as the recorder channel, he/she is referred to as the Recorder Participant. From time t_4 and on when the Recorder Participant is no longer involved in the call, CTI events are still received for that channel. This allows the system to supply information about the entire phone call involving extension 0001 that may be of interest to the customer.

32

Since the CRG must be prepared to handle messages from different components arriving in any order, it is designed to collect information in separate structures. Depending upon the operating mode of the CRG channel, call records are created from information collected in one or more of these repositories. The name given for these structures is Master Call Record (MCR).

The major components of the preferred embodiment contributing information for call records are the Recorder and the CTI Server. In alternate embodiments of the subject invention, other multimedia or screen image data may be provided to the CRG in order to be merged with descriptive metadata.

Recorder events are assembled into VOX MCRs identified by a unique sequence number. Individual events contain a sequence number identifying a specific structure to update (or create). For example, a recorder event would be used to indicate the beginning of a new audio segment. While that segment is active, other messages containing the same sequence number are used to add metadata to the audio segment. These update events include, without limitation: DTMF digit data; agent association information; change of audio filenames holding future audio data; selective record control; and ANI, ALI, DNIS information. DTMF is Dual Tone Multi-Frequency and refers to sounds emitted when a key is pressed on a telephone's touch-tone keypad; ALI is Automatic Location Identification, a signaling method that identifies the physical street address of the calling party and typically used to support Emergency 911 response. Finally, a disconnect message identifies the end of an audio segment.

Events received from the CTI Server are accumulated in CTI MCRs. Each event received from the CTI server contains a unique identifier. Events containing the same unique identifier are associated with the same CTI MCR. If any VOX MCR contains audio data that overlaps in time with Recorder Participants in a CTI MCR, then that audio data is transferred to the CTI MCR. If the absorption process causes all audio metadata for a VOX MCR to be consumed, the VOX MCR is deleted from the VOX list. Therefore, call records generated on the same channel will never have overlapping audio data. VOX MCRs containing leftover audio not absorbed by CTI MCRs are either be saved into the central database if of significant duration or discarded.

Data from a Master Call Record alone is processed into call record(s) that populate the system's central database. Thus, if the recorder channel is set up for VOX based recording only or if the CTI Server is down, VOX MCRs drive call record creation in the system. Otherwise, the CTI MCRs drive call record creation in the system.

The VOX and CTI MCR structures are maintained in two separate lists for each recording input channel. These are the VOX History List and CTI History List respectively. These lists represent a history of call activity sorted chronologically. The depth of the history list is driven by a configurable time parameter indicating the amount of history that should be maintained. By maintaining a history, the CRG tolerates events received in any order as long as received within the time boundaries of the history list. Some CTI Servers obtain data from SMDR type switches which report entire phone calls at the end of the call with a summary message. Maintaining a history buffer for VOX MCRs allows us to hold onto audio data for a period of time to allow later CTI summary messages to consume (absorb) the associated audio.

The MCR has status fields associated with them indicating its current state. At an installation involving real time

US 6,728,345 B2

33

CTI events, when a recording input channel receives a CTI event, it may indicate that a participant connected at the same telephony switch location as the recorder (Recorder Participant) is active in the call. The MCR is considered active as long as there is a Recorder Participant still active in the call. During this period, any new audio arriving on this channel is associated with the MCR. When a Recorder Participant leaves the call, the MCR becomes inactive. Since any Recorder Participant can become involved in the conversation at any given time through transfers or conferences, the MCR can transition into and out of active state many times throughout the phone call.

Another field in the MCR indicates the overall status of the call. This flag, called `m_bComplete`, indicates when the phone call is over. An MCR is considered incomplete as long as there is at least one participant still active in the call. When there are no participants active in a MCR it is considered to be complete. Therefore, calls created in real-time will start as incomplete and at some point transition into completed state. When an MCR enters complete state, a Closed Time variable is set to the current time. This time is used in maintenance of the History List. A closed MCR is allowed to stay in the History list for a configurable amount of time before it is deleted. During this window of time, events arriving out of timely order are allowed to update the MCR. Once this configurable amount of time expires, the MCR is updated in the local database, marked complete, and deleted from the History List.

When the CRG starts, it initializes, for each recording input channel, a location which identifies where it is attached to the telephony switch. Each recorder location contains status fields describing the state of the switch and CTI server involved. These fields are `m_SwitchStatus` and `m_MetadataServerStatus` respectively and are set to "down" state until an event is received that indicates otherwise. When a message is received indicating a change of state, all associated recorder locations are updated with the new state value. Any changes in operation are processed upon receipt of the next event for the channel.

Another configuration setting indicates what type of external sources are allowed to populate call records created on a record channel. This setting, `m_ExternMetaDataSource`, is set to zero when a record channel is to be driven by recorder events only. It is set to non-zero when external events are allowed to generate MCRs.

The CRG is able to react to a variety of situations that may arise. For example, when the CRG first initializes and a record channel is configured to receive CTI input, how are call records generated if the CTI server is not running? What if the CTI Server is running but the communication path to the recorder is down? The CRG must also be able to react to external parts of the system, that it normally relies on for input, being temporarily unavailable for periods of time. In accordance with a preferred embodiment, the CRG handles these situations by operating in different modes: Initial, Degraded, and Normal. These modes are applied individually to each channel in the recorder.

Initial Mode: When a recorder starts up, there can be a considerable amount of time before the rest of the system becomes operational. The CRG must be ready to handle events coming from the Recorder immediately after startup. Therefore, the CRG must be ready to accept recorder metadata without supportive information from the CTI server. VOX MCRs are created from these recorder events and are stored in the VOX History List. When VOX MCRs are completed, they are made persistent in the Local Data Store.

34

The CRG system will remain in this mode until all of the following conditions occur: (1) the CTI server becomes available; (2) the switch being recorded by this channel becomes available; and (3) a configuration option for the channel indicates it is to be driven from an online CTI server and switch.

Degraded Mode: If a record channel is configured to be driven from a CTI source, only CTI MCRs are entered into the database. These CTI MCRs absorb any recorder metadata that intersects with the time ranges of the CTI events. No VOX MCRs are made persistent. If, however, the CRG detects that the CTI Server, switch, or associated communication paths are down, the channel enters Degraded mode. This mode is similar to Initial mode in that VOX MCRs are made persistent when completed. Any CTI MCRs that were left open at the time the CTI Server went down are closed and updated for the last time. The recorder channel will remain in this state until the three conditions indicated in "Initial Mode" are met. Only then will the recorder channel transition into Normal mode.

Normal Mode: Under normal operating procedures in a system with a CTI server and switch online, MCRs are created whenever a VOX or CTI connect event is received and stored in the appropriate list. For each VOX message received, the CTI History List is swept to see if audio metadata can be absorbed by a matching MCR. Any remaining audio data is placed in a VOX MCR. For CTI events involving updates to Recorder Participants, the list of VOX MCRs is swept to see if audio metadata can be absorbed. CTI MCRs are made persistent to the Local Datastore when first created, upon significant update events, and when completed. VOX MCRs are not made persistent to the Local Datastore as they should be completely absorbed by CTI MCRs. There is a configuration parameter that can enable leftover VOX MCRs to be made persistent when they are removed from the VOX MCR history list.

Transitions from Initial/Degraded to Normal Mode: When a CRG channel is in Initial or Degraded mode, VOX MCRs are recorded into the Local Data Store when completed. If notification is received indicating a recorder channel meets the three criteria indicated in "Initial Mode", the channel is set to Normal mode. From this point on, only CTI based MCRs are made persistent and VOX MCRs will be absorbed by the VOX events. Since CTI events represent an accumulated history of a phone call, prior events occurring while the connection between the CRG and CTI Server was lost (or was not yet established) are nonetheless summarized in each update message. The time spans of Recorder Participant(s) are compared to audio data in the VOX MCR list, with any overlaps causing the audio data to be absorbed. In this way, any audio data that occurred while a connection to an external component is temporarily unavailable will still be capable of being correctly associated.

Transitions from Normal to Initial/Degraded Mode: When the CTI server and switch becomes available for driving the call record creation and processing, the CRG channel enters into Normal mode. A heartbeat message is used to periodically update the status of the switch and CTI Server. When the heartbeat is lost or there is a message indicating one of the components has gone down, the recorder channel switches to Degraded mode. The CRG will still create and maintain MCRs in the VOX list and force MCR closure on open CTI MCRs as they pass out of the CTI history buffer. The sweeping action of audio metadata among incomplete CTI MCRs will cease, preventing all future audio data from being absorbed by it. VOX MCRs are made persistent in the database when they leave the history buffer.

US 6,728,345 B2

35

Trunked Radio Mode: In an alternate embodiment of the subject invention, fields in the call record structure are added to support trunking radio. Information contributing to these fields may be obtained from communications with a Motorola SmartZone system. This system uses the Air Traffic Information Access (ATIA) protocol to communicate metadata related to radio activity. The embodiment has a trunking radio server similar to the CTI server that provides an interface between the SmartZone system and the recorders of the preferred system. This server provides the normalization of data and distribution to the correct recorder. There are currently two modes of operation of the Motorola trunking radio system that are discussed below.

Message Trunking: In this mode, when a radio is keyed, it is assigned a particular frequency to communicate on. When the radio is de-keyed, a message timeout timer (2–6 seconds) is started. If another radio in the talk group keys up during this time, the controller uses the same frequency for transmission and resets the timer. The conversation will remain on this frequency until the timer is allowed to expire. During this time, all events that are reported with respect to this conversation will have the same call number associated with them. Therefore, the concept of CTI based call records with many participants has been applied to Message Trunking.

If the timer is allowed to expire, future radio transmissions will be assigned to another frequency and call number. The server needs to detect this occurrence and properly terminate a call record.

Transmission Trunking: Transmission Trunking does not use the holdover timer mechanism used in Message Trunking. When a radio is keyed, it is assigned a particular frequency for transmission. When de-keyed, the channel frequency is immediately freed up for use by another talk group. Therefore, a conversation can take place over many channels without a call number to associate them. The concept of VOX based call records which contain one radio clip per MCR is used in this mode.

Selective Record: There may be certain phone calls involving extension or agents that are not to be recorded. Selective Record is a feature that tells the system to refrain from recording a call while a certain condition exists.

Virtual CRG: MCRs can exist in the subject system's database that have no audio associated with them. These non-audio MCRs can be created due to different features of the subject system. Some customers may require that all CTI data coming from their switch be saved even though they are not recording all extensions or trunk lines. By creating records from the CTI data alone, in the absence of recorded

36

audio, this mode of operation can provide the customer with useful information for statistical analysis or charting purposes. Likewise, records created based upon CTI data alone may provide a useful audit trail to verify the occurrence of certain telephone calls, analyze traffic patterns, or to perform other types of "data mining" operations. In that case, a CRG is associated with the CTI Server mechanism to receive all CTI events that are not matched to a specific recorder. These CTI MCRs are made persistent to the Central Database upon call completion.

Call Record Structure: Call record start and stop events originate from two independent sources: the Recorder and the CTI server. The CRG must perform some method of merging events from these two sources in such a way that the resultant call record contains the best information available. CTI server events are advantageous in that they provide more information than the recorder and can also accurately determine a call record boundary. Recorder based events are a subset of CTI server events and can only distinguish call record boundaries based upon VOX or off/on hook. The recorder has advantages in that since it is in the same box as the CRG, receipt of these events is guaranteed as long as the recorder is running. The main purpose of the assembly process is to leverage the information coming from the CTI server in such a way that the entire phone call is assembled into one Master Call Record (MCR). The structuring of call records is weighed towards trunk side recording with the services of the CTI server driving call record creation. This type of configuration enables the system to summarize phone calls in the most effective manner. The manner in which the structure of the MCR designed to achieve this goal is discussed below.

Master Call Record: The MCR holds information accumulated for all events received necessary for archiving to the local data store. It consists of individual fields that are global to the entire call record as well as lists of specific information. Global information includes identifiers for the call record, the start and stop times of the entire call, the recorder location with respect to the switch, and flags indicating the call record status.

Lists included with each MCR contain the following information: Media File List—List of media filenames that make up the call (e.g., telephone or radio communications); Screen Data Capture File List—List of screen image files associated with audio on this channel; and Participant List—List of participants involved in this call.

The MCR is populated from events received from the CTI Server and Recorders. The following table shows the fields in the MCR, in a preferred embodiment, their data types, description and if they are stored in the database.

Master Call Record structure.

Name	Type(max length)	Archive	Description
m_CallRecID	string†	Y	Unique ID (UUID) pertaining to entire call (CtI and Trunk Radio server provides same ID for call parts that are related to the same conversation.)
m_MetaDataSource	BYTE	Y	Indicates the source used to populate call record information. 0 = none 1 = CTI 2 = Trunking Radio

US 6,728,345 B2

37

38

-continued

Master Call Record structure.			
m_bCallComplete	bool	N	Indicates the end of a call. (i.e., there are no more active participants involved)
m_bCall-HoldoverExceeded	bool	N	If true, MCR has been in a complete state for a time period exceeding the configured Call Holdover time.
m_bMetadataHoldoverExceeded	bool	N	If true, MCR has been inactive for a time period exceeding the configured Metadata holdover period. Used to allow completion of MCRs that haven't been updated for long periods of time possibly because of missed events.
m_bLastUpdate	bool	N	true when the CRG has decided to send the last update of this MCR. Used to prevent any future updates.
m_bDontArchive	bool	N	Indicates whether this call record is to be archived by data store. Certain record features such as selective record may prevent us from storing this call record.
m_CallDirection	BYTE	Y	Indicates call origin Outbound = 0x12, Inbound = 0x21, Internal = 0x11, Unknown = 0x44
m_CustomerNumber	string†	Y	Variable length character array dedicated to information the switch may provide with the call. For custom call record support. (e.g., account number)
m_pRecLoc	RecorderLocation	N	Pointer to recorder location descriptor associated with this channel. (see RecorderLocation class)
m_SSFile	list†	Y	List of TimestampedFilename (see below) objects representing Screen Data Capture filename(s) associated with a call record
m_Participants	list†	Y	Array of CallParticipants (see below) describing all participants involved in the call
m_XactionSema	HANDLE	N	Semaphore used to lock this MCR from being modified by any other threads.
m_SemaTimeoutVal	unsigned long	N	Maximum time thread is blocked on m_XactionSema access before returning.
m_bModified	bool	N	Set whenever MCR is changed in a way that requires update to the Local Data Store.
<u>VOX Call Record (Derived)</u>			
m_dwVoxCrNum	DWORD	N	Sequence number of first VOX MCR associated with this OTI MCR (if applicable).
m_bVoxInProgress	bool	N	Indicates this VOX clip is still active (i.e., End time is default time.)
m_CreationTime	time_and_date†	N	Holds time at which the call record was created. Used for debugging purposes to measure how long a call record is alive.
m_CloseTime	time_and_date†	N	Local time at which MCR was marked complete. Used for determining when call record is ready for archive.
m_MediaFiles	list	N	List of TimestampedFilename classes representing multimedia files used to store data with respect to this call record.
m_CtiInfo	CtiInfo	N	Class containing CTI type data associated with call record.
<u>Base Call Record (Derived)</u>			
m_wVersion	WORD	N	Version number of call record.
m_StartTime	time_and_date†	Y	Start time of call record
m_EndTime	time_and_date†	Y	End time of call record

US 6,728,345 B2

39

40

-continued

Master Call Record structure.

RecorderLocation

m__MetadataServerStatus	BYTE	Indicates the status of the metadata server driving call records for this particular recorder location. This source is in most cases the CTI server but can be other servers such as Trunking Radio Server 0 = "down", 1 = "up"
m__SwitchStatus	BYTE	Indicates the status of the telephone switch providing call record information for this particular Recorder Location. 0 = "down", 1 = "up"
m__ExternMetaDataSource	BYTE	Indicates what external source (if any) is contributing call record meta data for this channel 0 = None (recorder only) 2 = CTI server
m__ChanID	ChannelIdentifier	Class identifying recorder channel.
m__SwitchID	Switch Identifier	Class identifying switch connection point.
m__SwitchChars	SwitchCharacteristics	Class identifying characteristics of switch needed by CRG.

SwitchIdentifier

m__SwitchNum	WORD	Number identifying switch
m__wTrunkID	WORD	Identification of trunk line attached to switch. (Valid only if not equal to -1)
m__dwVirtualChannel	DWORD	Identifies time slot of digital line (T1 or E1) of interest. (Valid only if TrunkID is not equal to -1)
m__Extension	string† (6)	Extension number (Valid only if m__wTrunkID equals -1)

ChannelIdentifier

m__wNode	WORD	Unique number used to distinguish between multiple Voice Servers.
m__wChannel	WORD	Unique number used to distinguish between multiple recording input channels within a Voice Server.
m__bSignalSupport	bool	Indicates if hardware associated with this channel supports on/off hook signaling.

SwitchCharacteristics

m__bTimeSynced	bool	Indicates if switch is synchronized with the system.
m__bRealTime	bool	Indicates if switch provides CTI info in realtime (true) or batched and sent periodically (false)
m__iCmdTimeOffset	int	Value that indicates any known time offset between events received at the switch versus the time the similar signal is received at the recorder. This value will be used to adjust CTI generated timestamps before comparing to recorder events

US 6,728,345 B2

41

42

-continued

Master Call Record structure.		
m_iSwitchTimeOffset	int	For switches that are not time sync'd with the system, this value indicates any known time offset between the switch and the system time. This can be utilized if has some way of updating the time delta between switches and our system on a periodic basis.
<u>CallInfo</u>		
RingLength	WORD	Time (in sec) between first ring signal and off hook.
DTMFCode	string† (50)	DTMF codes entered during conversation
Name	Type	Description
<u>TimeStampedFilename</u>		
m_APStartTime	Time_and__date†	Start time of audio file
m_StartTime	Time_and__date†	Start time of interest
m_EndTime	Time_and__date†	End time of interest
m_SegStartTime	Time_and__date†	Start time of segment inside file absorbed by this MCR.
m_SegEndTime	Time_and__date†	End time of segment inside file absorbed by this MCR.
m_PathName	string† (36)	Path describing the Voice Server and directory location where the audio files are located.
m_File Name	string† (36)	GUID-based name that uniquely identifies a specific audio segment's recording file.
m_wFileType	WORD	bitmap indicating types of media associated with MCR.
<u>bit Data</u>		
0 - Audio Present		
2 - FAX Present		
3 - Video Present		
m_wFileFormat	WORD	3 - Screen Capture data Present Recording format of media data, as defined by Microsoft Corporation's multimedia description file "mmreg.h"
m_bNew	bool	Used by local data store to indicate whether this record should be inserted (true) or updated (false) into the database.
m_bDiscard	bool	If true, don't allow playback or archiving of this media. Used for the Selective Record feature.
m_dwVoxCrNum	DWORD	Sequence number corresponding to VOX call record that provided this media.
m_iAssocPart	int	Index of Recorder Participant in the Participant list causing this media file to be associated with this MCR.
<u>CallParticipant</u>		
m_AgentID	string† (24)	Registered ID of agent at extension (CTI) or Radio Alias (Trunking Radio).
m_Number	string† (24)	Full telephone number of the participant (i.e., ANI, DNIS)
m_Console	string† (10)	Seating position of participant that can consist of one or more stations (CTI) or Talkgroup ID (Trunking Radio).
m_Station	string† (10)	Unique telephone set. Possibly with multiple extensions

US 6,728,345 B2

43

44

-continued

Master Call Record structure.		
m_LocRef	BYTE	Describes the location of participant with respect to the switch. (1 = internal, 2 = external, 3 = unknown)
m_SwitchLoc	SwitchIdentifier	Class identifying the position of a participant relative to the telephone switch.
m_StartTime	Time_and_date†	Time participant joined the call
m_EndTime	Time_and_date†	Time participant left the call
m_ConnectReason	BYTE	How participant joined the call NotConnected = 0, NormalStart = 1, ConferenceAdd = 2, TransferRecv = 3, UnknownConnect = 9
m_DisconnectReason	BYTE	How participant left the call NotDisconnected = 0, NormalEnd = 1, ConferenceDrop = 2, TransferAway = 3, OtherPartyHangup = 4, UnknownConnect = 9
Changed	Bool	Indicates if recent change in CTI message. (not archived)
<u>Trunking Radio Only Information</u>		
SourceSiteID	BYTE	Site number that is currently sourcing audio on active call.
ZoneID	BYTE	Zone at which participant is currently located.
CIUNumber	BYTE	Console Interface Unit. Translates 12kbit into clear audio & vice versa.
CDLNumber	BYTE	Channel associated with CIU
DIUNumber	BYTE	Digital Interface Unit. Translates ASTRO clear secure data into analog audio & vice versa.
DBLNumber	BYTE	Channel associated with DIU

†Objectspace data types

Unused string fields are null. Unused number fields are set to zero

The version number is used to indicate the structure of data contained within the call record.

In order to maintain compatibility with future versions, changes to call record structures will be performed in an additive nature. That is, current members of the call record will not change in position, size, or meaning.

Each call record will contain a list to store participant information. There will be at least two participants in a call record; the calling and called parties. Any additional connections that are conferenced in or transferred to are appended to the end of this list.

Only one active VOX and CTI based Master Call Record is allowed per recording input channel at any given time.

CRG Software Architecture

FIG. 18 shows the processing threads and data structures that comprise the CRG module in a preferred embodiment.

Event Processing: when the CRG is created and initialized, three threads are created. These threads are the CRG Event Processor thread 1810, Facade thread (The terms "facade," "facade," and "fascade" are used interchangeably in this disclosure) 1812 and Local Data Store thread 1816. Additionally, three message queues are created and are known as the Recorder 1824, Facade 1832, and Data Store 1844 queues, respectively. These queues enable the

processing of various input messages in a de-coupled fashion within the CRG, so that any delay in one area of communications does not affect the processing of another area. Each thread is described below.

Event Processor Thread: the Event Processor is the primary thread of the CRG module. Its responsibilities include reading any messages placed in the Recorder 1824 and Facade 1832 queues. The processing activities that occur in response to these messages cause updates to be made to call records belonging to one of the recording input channels 1856. If these changes cause a call record to be completed, a message is sent to the Date Store queue 1844 requesting that the call record be made persistent in the local database. This thread is also responsible for processing state change messages, that cause memory resident structures to be refreshed or to shut down the CRG module.

Facade Thread: The Facade thread handles messages that come from outside the Voice Server. Its primary function is to look for messages placed in the CRG's external Microsoft Message Queue (MSMQ) 1864 where events may arrive from other components within the overall subject system. Upon receipt of a message, the Facade thread reads the message, translates it into an appropriate format for the CRG's internal data structures, and places the translated copy in the Facade Queue 1832. This thread is known as the Facade, because it manages the external interactions of the CRG with the other components within the subject system.

Local Data Store Thread: The Local Data Store thread 1816 processes requests from the CRG Event Processor

US 6,728,345 B2

45

thread 1810. The primary purpose of the Local Data Store thread 1816 is to take internal Master Call Record (MCR) structures and translate their contents into structures compatible with database technologies, such as Microsoft SQL Server, or comparable types of storage means. These resultant structures are stored within the database in order to make the call record persistent.

Characteristics of some switches mandate that the CRG be able to handle CTI events that are not real-time. Some switches batch events and send them out periodically. CRG configuration settings that limit the history list by time must be set long enough to accommodate the switch characteristics. Therefore, call records that are generated between switch reports (via recorder events) will not be finalized until a configurable time period (window) after which the call record terminated. This window (CallHoldoverPeriod) needs to be set to a minimum of the period of time between switch reports. Once a call record leaves this time window, it is marked as read-only and committed to the local data store.

A situation that must be dealt with is when the telephone switch is not time synchronized with the rest of the system. To facilitate the merger of recorder and switch events effectively in non-time-synchronized systems, alternate embodiments of the subject system are described.

One alternate embodiment of the subject system has a mechanism that synchronizes the clocks in the system (manually or automatically) on a periodic basis. This must guarantee time skews of less than some small and known quantity. A second embodiment has a mechanism for measuring the time delta between the switch and the subject system. This value is updated periodically and used by the CRG during the merging process. A third embodiment implements a combination of the first two.

During the call record merging process, a global time delta is used to adjust switch event time stamps before comparing to existing call record data.

The following paragraphs define the types of events the CRG is designed to accept and process. These events may cause the CRG to initialize, process metadata into call records, or prepare the system for shutdown.

The Master Controller (a sub-component of the present system's Scheduling & Control Services) supplies system events. The Master Controller notifies the CRG of system related changes such as configuration changes, CTI server status and system shutdown events. The CRG changes its behavior based upon events received from the Master Controller.

System Events: The CRG provides an interface that allows the client application to control its states of operation. This is accomplished with an interface class that is used by most system components in the subject system. The interface is named IProcCtrl and supports the following methods: Initialize(); Start(); Stop(); Pause(); Resume(); Ping(); and Shutdown().

In addition to these methods, the CRG supports two event messages that inform it of status changes that are needed to either update its memory resident configuration information or change its mode of operation. These methods are CtiStatus and AgentExtensionStatus. Each method is described in the following paragraphs.

Initialization Event: This method is the first method that should be called after the CRG has been created. When the CRG object is created, it retrieves configuration information from the subject system's database. This information describes the number of channels in the recorder, the switch

46

location where each channel is connected, any fixed associations of telephone extensions or agent identifiers. Also included are parameters that determine the behavior of the CRG. Threads are spawned to handle the processing of CRG events, communicating with external metadata contributors, and processing information into the Call Records tables. These threads are created in a suspended state and require the Start or Resume commands to begin processing activity.

Start Event: This method should be called after the Initialization event. It resumes all threads of the CRG enabling it to process incoming events.

Pause Event: This method suspends all threads of the CRG.

Resume Event: This method is called after the Pause command to enable all CRG threads to continue processing.

Ping Event: This method is used by client applications to test the connection to the CRG. The method simply returns a positive acknowledgment to let the client know that the CRG is still running.

Shutdown Event: This method notifies the CRG when the subject system is shutting down so that it can cleanly terminate itself. The shutdown event supports a single parameter (ShutdownMode) that indicates how it should shutdown.

If the ShutdownMode is specified as "Normal", all pending events read from the input event queues and processed into the call records, any open call records remaining are closed at the current time and written to the database.

If the ShutdownMode is "Immediate", input event queues are cleared without processing into call records, open call records are closed and written to the database.

Once these actions are completed, the CRG threads terminate. At this point, it is now safe for the client application to release the resources of the CRG.

Stop Event: This method is implemented for consistency with the common interface of IProcCtrl. The CRG has no purpose for this method and just returns a positive acknowledgment.

CtiStatus Event: This event informs the CRG of the operational status of the CTI server that is providing it with telephony metadata needed for CTI call record generation. The Scheduler component of the subject system is responsible for maintaining a heartbeat with the CTI server to detect when connection has been lost. Any changes in CTI server status result in a CtiStatus message directed at the CRG.

This message contains one parameter that indicates the new state of the CTI Server. If the parameter indicates that a CTI Server has become operational, recording input channels associated with the CTI Server change from "Degraded" mode of operation of "Normal" mode. If the parameter indicates that the CTI Server is not operational, recording input channels associated with the CTI Server change from "Normal" mode of operation to "Degraded" mode.

AgentExtensionStatus Event: This event indicates that a change in one of the Agent or Extension tables has occurred. Since the CRG uses these tables to associate with recorder channels, the memory resident version must be updated. Therefore, this event causes the CRG to read these tables and update its memory resident copy.

Call Record Events: When a call record event is received, the message is interpreted to determine which recording input channel may be affected. Any filtering necessary on a per channel basis is performed at this stage. Call record

US 6,728,345 B2

47

events are then dispatched to the appropriate Call Record Channel Manager. There is a separate call record channel manager, which is a software sub-component of the CRG, for each recording input channel in a Voice Server. There are three messages that directly contribute to the creation and completion of call records. One comes from the CTI Server in the form of a CTI Event. The other two originate from the recorder and are the VoxSummary and VoxDisconnect messages. Each message is described in detail below.

CTI Event: The CTI Event is a message originating from the CTI Server software module that processes the information received from the telephone switch. The message details

48

each participant involved with the phone call as well as information global to the call such as ring duration and DTMF codes. A CTI event message is sent to the CRG whenever a change in participant status occurs as well as when new ones enter the call. The messages are cumulative in that all information of the previous messages is contained in the new one with any additions included. This makes for a more robust system in cases where one of the messages is lost.

The pseudo code for processing a CTI event is shown below:

Pseudo code for CTI Event

```
// ----- CTI Event (BEGIN) -----
// Don't process OTI events if we're not in correct mode
Is this recorder channel configured to receive OTI event data?
{ // Yes
  Does this event match an MCR in my CTI History list?
  { // Yes
    Update MCR participants with matching one in CTI event
    Add any new participants to MCR.
    UpdateMediaFiles() (see pseudo code)
  } // End - Does this event match an MCR in my history list?
  Otherwise
  {
    Create new MCR
    Initialize MCR Start time from Oldest Participant Start time in event
    Copy participants from event to message to MCR.
    // Now that we've updated the participants, see if
    // we need to change media file associations.
    UpdateMediaFiles() (see pseudo code)
    Insert new MCR into Cti MCR history list.
  }
}
Are there any participants still active?
  Mark MCR as active
Otherwise
  Mark MCR as complete
} // End - Is this recorder channel configured to receive CTI event data?
// ----- UpdateMediaFiles (BEGIN) -----
for each Recorder Participant in the MCR
{
  Is this not a new Recorder Participant?
  {
    // This participants start and/or end time may have been adjusted.
    // See if audio previously absorbed by it has to be returned to the VOX history
    list
    FindGiveBackMediaFiles() (see pseudo code below)
  }
  for each MCR in VOX History list with a time range that overlaps with this
  recorder participant
  {
    for each media file in this VOX MCR that's timespan overlaps with this
    recorder participant
    {
      CheckAndApplyMediaFile() (see pseudo code)
    }
  }
  }End - for each media file in this VOX MCR that's timespan overlaps with this
  recorder
  participant
  Did we consume all audio in this VOX MCR?
  Remove VOX MCR from History list and delete it.
}End - for each MCR in VOX History list that's time overlaps with this recorder
participant
}End - for each Recorder Participants in the MCR
GiveBackAudio() (see pseudo code)
// ----- UpdateMediaFiles (END) -----
// ----- FindGiveBackMediaFiles (BEGIN) -----
for each media file associated with the given CTI MCR
{
  Was this media file contributed from the given recorder participant?
  { // Yes
    if media file lies completely outside recorder participant timespan?
    |<-----Participant Timespan ----->| |<-----Media File timespan----->|
```

-continued

```
Pseudo code for CTI Event

Move this entire media file to the Giveback list
Otherwise, If media file start time is before the recorder participants start
time?
|-----Participant Timespan-----|
|-----Media File Timespan-----|
Make a copy of this media file and set its end time to the participants start
time.
Add media file to giveback list.
Set original media files start time to that of recorder participant.
Otherwise, if media file end time is after the recorder participants end time?
|-----Participant Timespan-----|
|-----Media File Timespan-----|
Make a copy of this media file set its start time to the participants end
time.
Add media file to giveback list.
Set original media files end time to that of recorder participant.
}
}End - for each media file associated with the given CTI MCR
//-----FindGiveBackMediaFiles (END)-----
// ----- GivebackAudio (BEGIN)-----
// Sweep through VOX MCRs re-populating any giveback audio
for all audio portions in given back list
{
if we find the VOX MCR this audio originally came from?
{ // Yes
Attempt to merge the give back media file with an existing VOX MCR media
file
that's start or end time is adjacent to this ones.
Otherwise, associate this media file with the VOX MCR.
}End - if we find the VOX MCR this audio originally came from?
Otherwise
{
// Original VOX MCR containing this audio file doesn't exist anymore.
Create a new MCR.
Associate the giveback media file with the new MCR
Insert MCR into VOX History list
}
}End - for all audio portions given back
// -----GivebackAudio (END)-----
// -----CheckAndApplyMediaFile (Begin)
Does Recorder Participant span the entire media file?
|-----Participant Timespan-----|
|-----Media File timespan-----|
Move media file from VOX MCR to Cti MCR.
Otherwise, Does Recorder Participant overlap with media file start time?
|-----Participant Timespan-----|
|-----Media File timespan-----|
Make a copy of this media file and set its end time to the participants end time and
associate with recorder participants MCR.
Set the original media files start time to the recorder participants end time.
Otherwise, Does Recorder Participant overlap with media files end time?
|-----Participant Timespan-----|
|-----Media File timespan-----|
Make a copy of this media file and set its start time to the participants start time and
Make another copy of this media file and set its start time to the participants end
time and associate with VOX MCR.
Set the original media files end time to the recorder participants start time.
// -----CheckAndApplyMediaFile (End)
```

VOX Summary Event: The VOX Summary Event is a message originating from the recorder associated with this 55
CRG. It can be used in one of two ways.

The primary use of this message is to indicate the start of audio activity in real-time. When used in this mode, the VOXSummary command indicates the beginning of audio activity. But since the activity is not complete, the end time 60
is set to indicate that the VOX segment is incomplete. The end time of incomplete media file is also set in this way. In this case, a VOX Disconnect message is required to complete the end times.

The second mode is used to indicate a history of audio 65
activity. The VOX Summary start and end times reflect the period of time covered by all accompanying media files. The

media files also have there respective start and end times filled in. This message is complete and thus requires no follow up messages. The VOXSummary message is shown below.

Field Name	Description
VOX Summary Message Format	
Channel	Recorder channel of audio activity
VOXCrNum	Sequence number used to correlate related VOX events.

US 6,728,345 B2

51

-continued

Field Name	Description
StartTime	time at which audio activity first started
EndTime	Time at which last audio activity ended.
Media Files	list of multimedia filenames used to store data with respect to this call record. (see below for details)
RingLength	Time from start of ring to off hook (in sec)
DtmfCodes	String of DTMF codes detected during VOXSummary period
ConnectReason	Indication of why VOX segment was started
DisconnectReason	Indication of why VOX segment was terminated
<u>Media File Structure</u>	
FileStartTime	Time corresponding to first byte of audio data in a file.
StartTime	Time corresponding to first byte of audio at which activity occurred
EndTime	Time corresponding to last byte of audio at which activity occurred
FileName	String containing name of audio file.
PathName	String describing the location of audio file.
iAssocPart	Used by the CRG to indicate with which Recorder Participant this audio segment is associated, when it is part of a CTI-based MCR.
dwVOXCrNum	Used by CRG to indicate which MCR in the VOX History list this audio segment originated.

The pseudo code for processing a VOX Summary event is shown below.

```
// ----- VOXSummary (BEGIN) -----
Is this recorder channel configured to receive CTI event data?
{ // Yes
  // Attempt to merge media files in message with CTI based
  MCRs
  for each CTI MCR in History list
  {
    If any of the given media files fall inside the timespan
    of the Cti MCR?
    { // Yes
      // Merge media files with overlapping recorder
      participants in CTI MCR
      for each given media file in VOX Summary message
      {
        for each recorder participant in the given CTI MCR
        {
          CheckAndApplyMediaFile () (see psuedo code)
        } End - for each recorder participant in the given
        CTI MCR
      } End - for each given media file
    } End - If any of the given media files fall inside the
    timespan of the Cti MCR?
  } End - for each CTI MCR in History list
  Remove media files from VOX Summary message that are
  completely consumed
}
Any unabsorbed audio remaining in message?
{ // Yes
  Create MCR for remainder of audio.
  Insert MCR into VOX History List
}
// ----- VOXSummary (END) -----
```

VOX Disconnect Event: The VOX Disconnect Event is a message originating from the recorder associated with this CRG. It is used to terminate a VOX segment that has been started by a real-time VOXSummary message.

52

The VOXDisconnect message is shown below.

<u>VOX Disconnect Message Format</u>	
Field Name	Description
Channel	Recorder channel of audio activity
VOXCrNum	Sequence number used to correlate related VOX events.
Time	End time of the VOX segment. Also indicates the end time of open media file.
DisconnectReason	Indication of why VOX segment was terminated

The pseudo code for processing a VOX Disconnect event is shown below.

```
// ----- VOXDisconnect (BEGIN) -----
Is there a MCR in VOX History list with the same sequence
number?
{ // Yes
  // Close and update all media files in both VOX and
  // MCR list related to this one
  Close Active media file in VOX MCR at given message time
  UpdateFromMediaFile ()
  // Update any CtiMCRs that absorbed the audio file
  closed.
  for each MCR in CTI History list
  {
    // Attempt to merge audio with MCR.
    // Look for matches with audio filenames.
    for each media file in Cti MCR contributed by this
    VOX clip
    {
      Close media file at given message time
      // Now that we've closed it, does this media file
      still
      // belong with this CtiMCR?
      Does media file still fall in time span of MCR?
      { // Yes
        CheckAndApplyMediaFile () (see pseudo code)
      }
      Otherwise
      {
        Remove media file from MCR list and discard
      }
    } End - for each media file in MCR contributed by
    this VOX clip
  }
  Close VOX MCR and mark as complete
}
// ----- VOXDisconnect (END) -----
// ----- UpdateFromMediaFile (BEGIN) -----
// Look for matches with audio filenames
for each media file in MCR contributed by this VOX clip
{
  Close media file at given message time
  // Now that we've closed it, does this media file
  still
  // belong with this CtiMCR?
  Does media file still fall in timespan of MCR?
  { // No
    CheckAndApplyMediaFile () (see psuedo code)
  }
  Otherwise
  {
    Remove media file from MCR list and discard
  }
} End - for each media file in MCR contributed by this
VOX clip
// ----- UpdateFromMediaFile (END) -----
```

US 6,728,345 B2

53

Data Events: Data events are appended to the currently open associated call record. For CTI data events, this pertains to a currently open MCR based upon CTI connect events and containing a matching call record ID. For VOX data events, the currently open VOX call record is affected. If an open call record doesn't exist, an error condition is reported.

Correction Events: Correction events exist to remove a previous alteration to a call record after it has already been populated. One reason for such an event is to support selective record. An audio file that cannot be recorded due to customer or legal reasons might need to be removed from the call record or the entire call record might need to be deleted. The VOX event for a filename might have already been processed into a call record before the selective record mechanism has determined it not to be recorded.

Selective Record (Exclusion): Selective Record is an important feature of the subject system, imposed by customer requirements. If the customer does not want certain participants recorded when they become involved in a recorded call, the CRG must exclude any audio associated with the call record for that participants' time of involvement. Implementing this feature is complicated by the varying characteristics of customer switches. If the telephone switch environments report events in real-time, recording of media can be prevented by turning the recording input channel off during the selective record participants' time of involvement. However, what happens when events are not reported in real time from the switch? The answer lies in the sweeping action of the CRG previously discussed for recorder participants.

The CTI Event message is routed through the Scheduler, and is altered by the Scheduler to indicate which participants re recorder participants as well as which ones are selective record participants. Recorder participants trigger the CRG to sweep any audio from VOX MCRs that overlap in time. When the CRG detects an overlap between recorder participant and selective record participant times, the audio that is swept into the CTI MCR for this overlap period is discarded. This causes the audio to be removed from both VOX and CTI MCRs, which prevents any chance of the audio being made available for playback or archive.

Selective Record Event: The Selective Record command is an event originating from the Scheduler. It identifies either a participant that is not to be recorded or that an entire call record should not be recorded. In one embodiment the system is capable of handling recording exceptions based upon information obtained from the CTI data. Criteria for selective record processing are discussed below.

Selective Record feature can take on two meanings. In one instance, a customer may want to record all telephony events except for ones that meet specific criteria. In a second instance, a customer may only want to record calls that meet certain criteria.

Since selective recording can possibly be triggered from multiple sources, in a preferred embodiment this decision process is located in the Master Controller, a sub-component of the subject system's Scheduling & Control Services.

54

Suggested reasons for not recording all or parts of a call are based upon the following examples of CTI event data.

Event Data	Explanation	Results
Agent exclusion based upon participants AgentID	Supervisor involved calls not to be included	Delete audio for agent's participation during the call, and the associated references in the MCR.
Exclusion based upon Extension or fully qualified phone Number of participant.	CEO involved calls not to be recorded. (whether at office or at home)	Delete audio for agent's participation during the call, and the associated references in the MCR.
Combination of AgentID of one participant and fully qualified phone number of another participant	Prisoner calls his lawyer.	Delete all audio as well as the entire call record.

Based upon these conditions and any future rules established inside the Master Controller (MC), exclusion can take place on audio recorded during a target participant's time of involvement or over the entire call record.

The chain of events involved in Selective Record (Call Exclusion) is as follows:

1. Recorder detects presence of audio and records to audio buffer.
2. Recorder sends VOX events to CRG indicating presence of audio.
3. CRG creates new call record based upon VOX event.
4. The CTI server sends call events to CRG and MC.
5. CRG associates CTI event data with VOX based call record.
6. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a Selective Record (exclusion) command is sent to both Recorder and CRG indicating the start of the selective record interval.
7. Recorder deletes audio indicated in selective record message and continues to suppress recording until instructed otherwise.
8. CRG alters the call record to eliminate details of participant or deletes the call record.
9. Upon completion of the call, the CTI Server sends call events to the CRG and MC.
10. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a selective record (exclusion) is sent to the Recorder indicating the end of the selective record interval.
11. The Recorder resumes its normal mode of audio recording.

Selective Record (Call Inclusion)

1. The CTI server sends call events to CRG and MC. CRG creates MCR and populates with events. Since default is set not to record, the flag `m_bDontArchive` is set to prevent the local data store from writing it to the database.
2. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a Selective Record (inclusion) command is sent to both Recorder and CRG indicating the start of the selective record interval. CRG sets `m_bDontArchive` to false and immediately instructs local data store to archive.
3. Recorder detects presence of audio and records to audio buffer.

US 6,728,345 B2

55

4. Recorder sends history of VOX events to CRG in a VoxSummary message.
5. CRG creates new call record based upon VOX event.
6. CRG associates CTI event data with VOX based call record.
7. Upon completion of the call, the CTI Server sends call events to the CRG and MC.
8. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a selective record (inclusion) command is sent to the Recorder indicating the end of the selective record interval.
9. The Recorder resumes its normal mode of suppressing the audio recording.

The format of the Recorder's Selective Record command is shown below.

Name	Type	Description
StartTime	time_and_date	Start Time of recording interval
EndTime	Time_and_date	End Time of recording interval
bRecordAudio	bool	If true, record audio during the indicated interval. If false, suppress any audio recording during the indicated interval.

Since the recorder has no knowledge of participants or call record boundaries, the MC needs to inform the recorder when to start a selective record interval and when to stop. The boolean bRecordAudio signifies what action should be taken during this interval.

When an event occurs that triggers the start of a selective record interval, the Recorder's selective record command informs the recorder of the interval start. The End time is most likely not known at this point so it is set to some invalid value in order to indicate that audio should be recorded (or suppressed) for an indefinite period until a subsequent command is received.

When an event occurs that triggers the end of a selective record interval, the Recorder's selective record command informs the recorder of the interval end. The End time indicates when the selective record interval is complete. The recorder returns to its normal recording mode based upon its original configuration.

Any selected audio committed to file needs to be removed from the file and replaced with a silence entry for that period.

The format of the CRG selective record command is shown below.

Name	Type	Description
MCR Number	UUID	MCR affected by this selective record command
Participant Index	UINT	Index of participant in MCR not to be recorded. (if Reason=1)
Reason	BYTE	1=Participant, 2=Entire Call

For the CRG, only a single event that indicates what is selectively recorded is needed. If the Reason code indicates that the entire call record is to be deleted, the CRG will mark the call record such that it is removed from the database if it has already been written or not logged in the first place. If selective record affects a specific participant, the call record

56

can either be left unmodified (since the recorder has already handled deletion of audio) or the participant can be overwritten to remove his/her details.

The system configuration can be adjusted so that the CRG will operate in either fashion, depending on whether removing the audio alone is sufficient for the desired application of the system, or if the metadata must also be removed to eliminate the records of telephone numbers dialed, etc.

CRG Software Implementation

In the preferred embodiment of the subject system, the CRG is implemented as an in-process COM DLL that is associated with the Audio Recorder process, and therefore these two components reside together upon the Voice Server. COM, here, is Common Object Model, a distributed computing architecture designed by Microsoft Corporation to facilitate cooperative processing among software elements on a LAN. DLL is Dynamic Link Library, a means whereby executable code can be encapsulated in a package that can be loaded upon demand and shared by several programs, rather than being packaged as a separate, isolated executable program. The Audio Recorder process is responsible for creating the CRG COM object as well as starting and stopping the CRG subsystem. The Data Store module that interfaces with the CRG is a statically linked DLL.

Class Design

FIG. 19 illustrates the class diagram of the Call Record Generator. The CRG module is itself comprised of a plurality of modules, as shown in the figure, and explained below.

CallRecordEvent Processor—the CallRecordEventProcessor class 1912 is the main class of the CRG. It is instantiated during the Initialize method call of the CRG interface. It is responsible for allocating the rest of the CRG objects. On instantiation, it acquires the channel count for the recorder (currently limited to 128) and instantiates a group of classes for each recording input channel. These classes include a CallRecordChannelManager 1916 and RecorderLocation 1920 for each channel. The CallRecordEventProcessor 1912 creates the Recorder 1924 and Facade 1928 Event input queues. Reading and processing of configuration information from the subject system's database takes place in the CallRecordEventProcessor 1912. Events received that cause a change in configuration are processed there.

CallRecordChannelManager—This class manages the call records for a specific recording input channel. It is responsible for creating, populating, and closing call records with event information received from the CRG event processor. If event information is deemed as significant, the CallRecordChannelManager 1916 will send an event to the DataStoreEventQueue 1932 in order for the update to be reflected in the local data store.

MasterCallRecord—This class 1936 holds information that is global to an entire call. Global information includes identifiers for the call record, the start and stop times of the entire call, the recorder location with respect to the switch, and flags indicating the call record status. It also contains a list of the participants within a call, based upon information supplied by CTI events. It acts as a centralized point of control for merging call record information for a given telephone call.

VoxCallRecord—This class 1940 is a superclass of the MasterCallRecord class 1936. It contains information dealing with events provided by the recorder. It holds the details

US 6,728,345 B2

57

of a call, such as the start/stop times, media filenames and other data that can be supplied by the recorder.

RecorderLocation—This class 1920 holds the information relating a logical device on a telephony switch with a specific Voice Server and recording input channel.

The following table indicates configuration information needed by the CRG at runtime.

Configuration Field	Type	Acceptable Values	Default	Description
sSysTimeCoupling	String	"TIGHT", "LOOSE"	"LOOSE"	Indicates how time-based recorder and CTI events are compared to determine a match. TIGHT - Recorder times must fit entirely inside CTI times for a positive result. LOOSE - Recorder times need to overlap with CTI times for a positive result.
nCompleteCallHoldOver Period	DWORD	0..4294967296 (in sec)	90 - For realtime CTI. (Much larger if non-realtime CTI)	Maximum number of seconds that a completed call record is kept in the history list. This holdover allows events coming from different sources to affect the call record before it is made persistent. After this holdover period expires, no more events can update the call record.
nActiveCallHoldoverPeriod	DWORD	0..4294967296 (in sec)	86400 (24 hours)	The maximum number of seconds a call is allowed to exist before being forcibly closed. This is used as a safeguard against missing CTI or Recorder events that would normally end a call record.
nMCRMaxSize	WORD	0..65535 (in entries)	100	Maximum number of entries allowed in the MasterCall Record history list
nSystemSkew	WORD	0..65535 (in sec)	0	A known, fixed difference (in seconds) that specifies the skew between a Recorder clock and a PBX clock. Used to adjust incoming CTI event times before processing and comparing with Recorder event times.
ynCTIDataFromRecorder	bool	1 = yes 0 = no	yes	Identifies, in cases where Recorder and CTI information overlaps, which source is preferred to populate the call records.
nSaveVoxClipsLonger ThanSeconds	WORD	0..65535	6	This setting is used to avoid creating VOX based call records from noise on recording input channels. It directs the CRG to discard any VOX clips that do not exceed the specified number of seconds in duration.

58

As discussed above, CTI is provided through a data link from specific telephone switching equipment located at the customer site, which is then input to the recording system's CTI Server. Supplied data includes such items as telephone numbers of involved parties, caller ID/ANI information, DNIS information, and agent ID numbers. The CTI Server performs the task of analyzing and reorganizing data from both the real-time and SMDR (asynchronous) links, and

Stream Control Manager

As noted above, in a preferred embodiment, the system of the present invention taps into activity on a PBX (Private Branch Exchange) by intercepting audio on either the trunk or extension side of a phone call. The tapped audio is then redirected as input to a channel on a DSP (Digital Signal Processor) based voice processing board, which in turn is digitized and stored into program-addressable buffers. The recorded audio is then combined with descriptive information ("metadata") obtained through a Computer Telephony Integration (CTI) communications link with the PBX and stored as a single manageable unit ("Voicedata") to facilitate its subsequent search and retrieval.

The preferred embodiment leverages Computer Telephony Integration, to supplement the recorded audio data.

passing the results onwards to the remainder of the recording system for further processing.

A module called the "Call Record Generator," or CRG, discussed above, is then responsible for collecting data from the CTI Server, creating 'master call records' and attempting to match those records with existing recorded audio data. If the CRG receives CTI information indicating that audio data recorded on two Voice Servers is related (for example, due to a transferred call), records will be generated for each portion with a common call record ID. This ID can later be used to query for all the pieces (or "segments") comprising the complete call. In addition, each segment will indicate the Voice Server which contains that piece of the call.

During playback, the User Workstation's player module connects to a program located on a Voice Server called the Playback Server, or PBServer. The machine name of the

US 6,728,345 B2

59

particular Voice Server with which a communications session should be established, stored by the CRG in the call record table of the Voicedata storage module, is passed into the player module after being extracted by the User Workstation's call record browser. A call record playback request is then submitted, which causes the PBServer to query for a specific call record's audio files located on that physical machine, open them, and prepare to stream the audio upon buffer requests back to the client. If successful, a series of requests is then issued from the client, each of which will obtain just enough audio to play to a waveOut device while maintaining a safety net of extra audio in case of network delays. Upon a request to "Move" within the scope of a call record, the PBServer will reposition its read pointer to the desired location and then begin passing back buffers from that point. This series of Request and Move commands will continue until the user chooses to end the session by shutting down the client-side audio player.

When a call is transferred between locations, it is possible that the call may span multiple Voice Servers, since the extensions or trunks involved may be monitored by different recorders. If this is the case, the audio data is spread out between playback servers, and it must be properly pieced back together to reconstruct the complete call for a playback client.

There are several possible solutions to the problem. First of all, one could choose one central server and copy in all data from the involved servers. This is as slow as copying the files locally to the client, but it at least consolidates the data to one location for the playback server to operate on. Assuming that this method is chosen, however, several new problems arise. First is the issue of drive space: depending on the number of transfers and recorders involved with a call record, the central playback server could end up suddenly storing a large number of files. This is multiplied by the total number of clients requesting playback sessions. Soon enough, a large amount of unpredictable space is being allocated and freed without any reasonable way of estimating the space necessary to service all requests. Similarly, the processor and memory load on this server is taking the brunt of being used for every playback request, since even normal, single recorder playback sessions would be routed through this one machine.

Another solution would be to have the central playback server run some intermediate process that would stream all of the data from the multiple servers back to each client, like a "funnel." This would avoid the copying and drive space issues, but there are still two problems. First, the centralizing of this server once again puts the entire load on a single machine. But more importantly, if multiple streams are being funneled through this one location, the server would somehow need to organize the streams so that during playback, they appear to be arranged in the proper order.

The Stream Control Manager (SCM) used in accordance with a preferred embodiment is the result of addressing the issues referred to in the second solution discussed above. With regard to the resource issue, the solution was to simply move the "funneling" module from one central server to the client side. In this way, servers are still providing the actual requested data, but it becomes the client side's responsibility to bring the data together. Yet the SCM remains a separate, COM-based module so encapsulation is still maintained (a client application is not hard-wired directly into the SCM code). This was intentional since other system modules in alternate embodiments of the system need to reuse the SCM to gather playback data (e.g., for phone handset playback support instead of LAN playback support) or to gather audio

60

from a multitude of Voice Servers for long-term offline storage on DAT or DVD media.

The process of stream management begins when the SCM is sent a list of segments which comprise the entire call. Each segment includes the machine name of the Voice Server, the segment's start time, duration, channel ID, and an event callback routine provided by the client which serves as a destination for the final organized data.

Once this list is received and stored as a vector (array), the SCM proceeds to try connecting to all servers required to play back this call. The connection, if successfully established, is associated with its respective segment via a pointer in the segment entry. The connection is also added to an array so that if a subsequent segment's server is the same as an earlier segment, the connection can be reused. This may occur if a call transfers away to a line monitored by a second recorder and is later transferred back again to the original line. If the process cannot complete successfully (i.e., if a Voice Server is malfunctioning), playback is aborted to avoid skipping over any necessary data.

Next, the SCM goes through its list of segments and for each, handshakes with its server through a series of function calls. During this phase, the SCM informs each playback server of the desired segment to stream back by providing its start time, duration, channel ID using the parameter data that was passed in earlier. Once again, if any part of the procedure fails, the entire initialization (and thus playback) is aborted. At the completion of this phase, every server should have loaded all the audio files associated with their portion of the entire data stream. Each is now ready for audio buffer requests.

The SCM then waits for a client to execute a "Start-Stream" call. In a graphical interface, this would occur, for example, when a user hits a Play button or begins a Save operation. Once this function is called, a separate thread spawns which will handle the entire process.

First, the current play position is checked to see which segment to begin playing on (a Move operation, explained below, controls the manual repositioning of this value). This is determined by looping through all of the segments, adding each segment's duration to a running total. When the current segment's duration added to the total exceeds the play position, that is the segment which contains the current play position.

Once this calculation is complete, a loop begins which starts from the previously determined segment and proceeds through the rest of the segment vector. For each segment, requests are formed for a predetermined buffer size and sent to the associated server. Once a buffer is returned, based on a flag configurable from the client, the SCM will either directly send back this data or "slice" it for the client first before returning it. Here, slicing refers to a process of dividing the buffer into smaller buffers by a least common multiple known as a block align; this is sometimes useful to a client with a graphical component because the interface may need to reflect the amount played in smaller subdivisions.

When it is detected that all data from a segment has been requested, the SCM automatically steps to the next segment (possibly located on a different Voice Server) and begins requesting data from it instead. Because all Voice Servers are pre-loaded with the data and "ready to go," this process takes place in a fraction of a second, and the client does not sense any gap in the audio data being returned. In fact, the only true method for discerning the segment boundaries involves listening for normal, audible indicators of a transfer

US 6,728,345 B2

61

being made (clicking, ringing, or hearing the voice of a new participant) as provided through the telephone switch environment.

At the close of a play session (e.g., the user hits Stop or Pause in a typical audio playback GUI displayed in conjunction with the GUI described in FIG. 16) a StopStream call is made to the SCM. The thread in turn detects that the stopped state has been entered, exits from the request loop code, and frees up any used resources. Finally, it informs the client that a Stop event has occurred. If the entire call record is played without calling StopStream, the SCM performs the same exit and cleanup code, but informs the client that a Done event has occurred instead.

Movement within the overall stream is straightforward, given the aforementioned method that the SCM uses to determine which segment to begin playing from. A global variable holds the total number of milliseconds of audio data requested thus far. When a Move is performed, the server containing the data at the destination position is told to re-position itself, and the current play position is reset. Now, once StartStream executes again, it will initially start requesting from the server that was just moved to. And because that server had also moved its position pointer ahead, data will not be streamed from the beginning of the segment, but from where the Move position fell within that segment. Thus movement is a synchronized action completely transparent to the client, who is, ultimately, only interested in treating the data as a single stream.

SCM Pseudo-code

1. Initialize receives segment description data (start time, duration, etc.)
 - a.) Form a vector of all segments.
 - b.) Try to connect to all segments' servers.
 - c.) If there is an error connecting to any server, exit.
 - d.) Try to initialize each connected server.
 - e.) If there is an error initializing any server, exit.
 2. If StartStream received:
 - a.) Go through segment list. Find segment of current play position.
 - b.) Starting with that segment, contact the associated server and begin requesting buffers.
 - c.) If option set, divide up buffer into smaller chunks.
 - d.) Send buffer(s) to client via event callback.
 - e.) Repeat until all data requested for this segment on that server.
 - f.) Repeat from step b. with next segment in list.
 3. If Stop received:
 - a.) Exit from request loop.
 - b.) Clean up used resources.
 - c.) Send "Stop" event back to client.
 4. If Stop not received, but all data from all segments played:
 - a.) Exit from request loop.
 - b.) Clean up used resources.
 - c.) Send "Done" event back to client.
 5. Move received:
 - a.) Go through segment list. Find segment of desired play position.
 - b.) Contact the associated server and reposition to that desired position.
 - c.) Reset current play position variable to reflect change.
- Detailed flow diagrams describing SCM operation are provided in FIGS. 20, 20A, 20B, 21, 22, 22A, 22B, and 22C.

FIG. 20 illustrates the initialization process of the Stream Control Manager. The Initialization Sequence begins when

62

a user enters the User Workstation playback software and at step 2010 queries for a recorded call record by desired criteria. At step 2012 a call record browser displays resulting call records. At step 2014 the user selects the desired record for playback. At step 2016 the browser invokes a PbkControlWin object: a dialog containing the 'player' ActiveX control.

At step 2020 the browser sends information to PbkControlWin about all segments comprising the call record. If at step 2024 immediate playback is not required, at step 2028 the entry is added to a playlist for future playback, and at step 2030 SUCCESS is returned. If at step 2024 immediate playback is required, at step 2032 the call record ID and segment list are forwarded to a GUI Player module. At step 2038 (see FIG. 20A) the player module instantiates a local SCM (StreamControl) object and stores a pointer in m_plStreamControl. At step 2040 the player module accepts the data, displays starting time and total duration (by parsing out string data), and forwards it to the final module, the Stream Control Manager (SCM), for audio playback.

Step 2046 begins the creation of a segments vector. At step 2046, a segment is parsed out from segList. At step 2048, recorder ID, start time, duration, and channel are parsed out from the segment. At step 2050, a new SEGMENT structure is created from recorder ID, start time, duration, and channel. At step 2052, a new SEGMENT is added to the SEGMENT vector. At step 2054, if all segments have been parsed from segList, at step 2058 an element is gotten from the SEGMENT vector. If at step 2054 more segments remain to be parsed from segList, steps 2046, 2048, 2050, and 2052 are repeated.

After step 2058, the program determines at step 2060 whether a new DCOM connection is required to the recorder for this segment. If not, at step 2062 the existing pointer is copied from the Connections vector to the server pointer in the SEGMENT vector and the program proceeds to step 2076. If at step 2060 the connection is new, a connection is made to the indicated recorder's "PlayBackServer" DCOM object using CoCreateInstanceEx. At step 2066 the program checks whether the object instantiated successfully. If not, at step 2068 a log error message occurs and at step 2070 ERROR (C) is returned. If at step 2066 the object instantiated successfully, at step 2072 (see FIG. 20B) the new object's pointer is added to the Connections vector. At step 2074 the program determines whether all segments have been connected. If not, the program returns to step 2058. If at step 2074 all segments have been connected, at step 2076 an element is gotten from the SEGMENT vector. At step 2078 the program queries for a list of wave files on the server that go with this segment. At step 2080 the program determines whether the query was successful. If not, at step 2082 a log error message occurs, and at step 2084 ERROR (C) is returned.

If at step 2080 the query was successful, at step 2088 the program opens the wave files on the server and prepares them for streaming. It also returns the wave format of the audio in the segment. At step 2093 the program determines whether the wave files and format were obtained successfully. If not, at step 2094 a log error message occurs and at step 2095 ERROR (C) is returned. If step 2088 is determined at step 2093 to have been successful, at step 2096 the program checks whether all segments have been initialized. If not, the program returns to step 2076. If so, step 2097 is performed and at step 2098 SUCCESS is returned.

FIG. 21 illustrates how the program manages a Player Object 2110 and a PbkControlWin Object 2132.

FIG. 22 illustrates the playback sequence of the Stream Control Manager. Initially, at step 2202 a user has completed

US 6,728,345 B2

63

initialization and is waiting to hit Play in the Player GUI. At step 2204 the user hits the Play button. At step 2206 a message is sent to the Play method in the Player ActiveX control. At step 2210 the Play method in Player ActiveX control causes the output buffers to be "sliced" to increase the number of smaller buffers sent, thus increasing the resolution of the "totalPlayed" variable. At step 2218 Play method causes the server-side position to move to the current slider position. At step 2222 the program gets segment i++ from the SEGMENT vector. At step 2224 (see FIG. 22A) the program determines whether the End Time offset for segment i is greater than curPosition. If not, the program returns to step 2222. If so, the program proceeds to step 2226 and causes the file pointer on the server side to change to the appropriate new location. The program checks at step 2230 whether step 2226 was successful. If not, at step 2232 a log error message occurs and at step 2234 ERROR (C) is returned.

If at step 2230 step 2226 is determined to have been successful, at step 2238 the program calls Stream Control::StartStream. At step 2242 the program gets segment i++ from the SEGMENT vector. At step 2244 the program calls CoMarshalInterThreadInterfaceInStream to marshal a DCOM pointer member across a thread boundary. At step 2246 the program determines whether all SEGMENT elements have been marshaled. If not, the program returns to step 2242. If so, at step 2248 the main SCM streaming thread is spawned.

FIG. 22B illustrates an SCM main streaming thread. When the thread begins, at step 2250 the thread gets a segment from the SEGMENT vector. At step 2252 CoCetInterfaceAndReleaseStream is called to unmarshal a DCOM pointer member across the thread boundary. At step 2254 the thread checks whether all SEGMENT elements have been unmarshaled. If not, the thread returns to step 2250. If at step 2250 all SEGMENT elements are determined to have been unmarshaled, at step 2256 the thread gets a segment from the SEGMENT vector. The thread then checks at step 2258 whether the End Time offset for segment i is greater than curAmountRequested. If not, the thread returns to step 2256. If so, at step 2260 the thread gets Segment[i++]. The thread checks at step 2262 whether i is less than the highest segment number. If not, an Event::Done method is called at step 2264, and at step 2266 SUCCESS (C) is returned. If so, at step 2268 the thread determines whether this is the first segment to be played in this instance of the thread. If not, at step 2270 the thread calls PBServer::PositionPLay(totalRequested) for Segment[i] and goes to step 2272. If so, the thread goes directly to step 2272.

At step 2272, the thread checks whether totalRequested is less than Segment[i].endTimeOffset. If not, the thread returns to step 2260. If so, the thread proceeds to step 2274 and checks whether totalRequested plus bufferSize is less than or equal to Segment[i].endTimeOffset. If not, at step 2276 the thread calculates a new bufferSize in multiples of the audio format's "block align" and proceeds to step 2278 (see FIG. 22C). If so, the thread proceeds directly to step 2278. At step 2278, the thread calls PBServer::ReqBuffer for Segment[i]. This is the core routine that actually retrieves a buffer of data from the PlayBack Server. At step 2286 the thread checks whether step 2278 was successful. If not, at step 2284 a log error message occurs, and at step 2282 ERROR (C) is returned.

If at step 2286 the thread determines that step 2278 was successful, at step 2287 totalRequested is set equal to totalRequested plus Actual returned buffer size. At step

64

2288, the thread checks whether Blockslicing is enabled. If not, at step 2289 the thread sends the buffer back to the Player via Event::SendData method and returns to step 2274. If BlockSlicing has been enabled, at step 2292 the thread checks whether the CODEC is Dialogic OKI ADPCM or PCM. If not, at step 2293 the slice of the slices is set equal to the audio format's block align and the thread proceeds to step 2296. If so, at step 2294 the size of the slices is set to an even dividend of the buffer size (e.g., one-tenth of the buffer size). At step 2296, the thread copies out "slice size" from the buffer and sends it back to Player via Event::SendData method. At step 2298 the thread checks whether the entire buffer has been sent back. If not, the thread returns to step 2298. If so, the thread returns to step 2274.

The Stream Control Manager could theoretically be adapted to be used in more general streaming media situations, outside that of communications recording systems. In most current stream-based systems for network-based playback of audio content, such as RealMedia and NetShow, two general broadcast architectures exist known as unicast and multicast. Unicast involves a single client-server connection for data streaming, while in the multicast scenario a server pushes data to a single network address which multiple clients can then "tune in" to. However both models assume that data is being continuously fed from a single server. In the interest of load balancing, or if pieces of a streaming presentation were spread out across multiple locations, the SCM model could provide an innovative solution where the client side has the power to weave together many streams into a single playback session. An example could be imagined where a news organization, such as CNN, dynamically assembles a streaming broadcast for the online viewer from many different reports located on servers across the country. The components could be played seamlessly end-on-end using the SCM model, and if the viewer desired to rewind or fast-forward to a specific point in the stream, the SCM model would allow for complete transparent control.

The present invention is not to be limited in scope by the specific embodiments described herein. Indeed, modifications of the preferred embodiment in addition to those described herein will become apparent to those skilled in the art from the foregoing description and accompanying figures. Doubtless, numerous other embodiments can be conceived that would not depart from the teaching of the present invention, which scope is defined by the following claims.

All the features disclosed in this specification (including any accompanying claims, abstract, and drawings) may be replaced by alternative features serving the same, equivalent, or similar purpose, unless expressly stated otherwise. Thus, unless expressly stated otherwise, each feature disclosed is one example only of a generic series of equivalent or similar features.

What is claimed is:

1. A system for recording information regarding telephone calls with three or more participants and comprising one or more telephone call segments, comprising:

- (a) a first memory having one or more locations storing audio data of telephone call segments
- (b) a second memory having one or more locations storing data regarding telephony events associated with the telephone call segments; and
- (c) a processor programmed to:
 - (i) identify telephone call segments that relate to the same telephone call and
 - (ii) construct a data representation of lifetimes of the telephone calls that have three or more participants,

US 6,728,345 B2

65

wherein said data representations are constructed using data regarding telephony events associated with the telephone call segments.

2. The system of claim 1 wherein the data representation of each telephone call comprises

- (i) a list of participants in the telephone call;
- (ii) a list of telephony events regarding the call;
- (iii) a list containing the time each telephony event occurred; and
- (iv) the start and end time of the call.

3. The system of claim 1 wherein the data representation of each telephone call comprises, for each segment of the call, the location of the stored audio data of that segment.

4. The system of claim 1 wherein the first memory and the second memory are the same.

5. The system of claim 1 wherein the processor is comprised of a plurality of physically separated components.

6. The system of claim 3 wherein the location of the stored audio data of each segment comprises a location of a .WAV file containing the audio data.

7. The system of claim 6 wherein the data representation of a telephone call further comprises an offset within the .WAV file to the start of the stored audio data.

8. The system of claim 1 wherein the data regarding telephony events is received from a plurality of sources connected to a telephone switching environment.

9. The system of claim 1 further comprising display software that uses said data representation to display a graphical representation of said telephone call.

10. The system of claim 2 further comprising display software that uses a data representation of a telephone call to display a graphical representation of said telephone call.

11. The system of claim 10 wherein the graphical representation comprises a representation of each segment of the call.

12. The system of claim 10 wherein the graphical representation comprises a representation of the length of time of each segment of the call.

13. The system of claim 9 wherein the display software further displays a table comprising data from the data representation.

14. A method for recording information regarding telephone calls with three or more participants and comprising one or more participants and comprising one or more telephone call segments, comprising:

- (a) receiving audio data regarding one or more telephone call segments;
- (b) receiving data regarding telephony events associated with said telephone call segments;
- (c) storing the received audio data regarding telephone call segments;
- (d) storing the received data regarding telephone events associated with said telephone call segments;
- (e) identifying telephone call segments that relate to the same telephone call; and
- (f) constructing data representations of lifetimes of telephone calls, wherein said data representations are constructed using data regarding telephony events associated with telephone call segments.

15. The method of claim 14 wherein each data representation of a telephone call comprises:

- (i) a list of participants in the telephone call;
- (ii) a list of telephony events regarding the call;
- (iii) a list containing the time each telephony event occurred; and

66

(iv) the start and end time of the call.

16. The method of claim 14 wherein each data representation of a telephone call comprises, for each segment of the call, a location of stored audio data if that segment.

17. The method of claim 14 wherein the received audio data and the data regarding telephony events are stored in the same memory.

18. The method of claim 14 wherein each data representation is constructed by a plurality of physically separated processors.

19. The method of claim 16 wherein the location of the stored audio data of each segment comprises a location of a .WAV file containing the audio data.

20. The method of claim 19 wherein a data representation further comprises an offset within the .WAV file to start of the stored audio data.

21. The method of claim 14 wherein data regarding telephony events is received from a plurality of sources connected to a telephone switching environment.

22. The method of claim 14 further comprising the step of using a data representation of a telephone call to display a graphical representation of the telephone call.

23. The method of claim 15 further comprising the step of using said data representation of a telephone call to display a graphical representation of the telephone call.

24. The method of claim 23 wherein the graphical representation comprises a representation of each segment of the call.

25. The method of claim 23 wherein the graphical representation comprises a representation of the length of time of each segment of the call.

26. The method of claim 22 further comprising the step of displaying a table comprising data from the data representation.

27. A system for recording information regarding telephone calls comprising one or more telephone call segments, wherein said calls comprise calls wherein at least one participant participates in a plurality of segments, comprising:

- (a) a first memory having one or more locations storing audio data regarding telephone call segments;
- (b) a second memory having one or more locations storing data regarding telephony events associated with telephone call segments; and
- (c) a processor programmed to:
 - (i) identify telephone call segments that relate to the same telephone call;
 - (ii) identify multiple call segments that have the same participant; and
 - (iii) construct data representations of lifetimes of telephone calls using data regarding telephony events associated with telephone call segments.

28. The system of claim 27 wherein a data representation of a telephone call comprises:

- (i) a list of participants in the telephone call;
- (ii) a list of telephony events regarding the call;
- (iii) a list containing the time each telephony event occurred; and
- (iv) the start and end time of the call.

29. The system of claim 27 wherein each data representation of a telephone call comprises, for each segment of the call, a location of the stored audio data of that segment.

30. The system of claim 27 wherein the first memory and the second memory are the same.

31. The system of claim 27 wherein the processor is comprised of a plurality of physically separated components.

US 6,728,345 B2

67

32. The system of claim 29 wherein the location of the stored audio data of each segment comprises a location of a .WAV file containing the audio data.

33. The system of claim 32 wherein a data representation of a telephone call further comprises an offset within the .WAV file to the start of the stored audio data. 5

34. The system of claim 27 wherein data regarding telephony events is received from a plurality of sources connected to a telephone switching environment.

35. The system of claim 27 further comprising display software that uses a data representation of a telephone call to display a graphical representation of said telephone call. 10

36. The system of claim 28 further comprising display software that uses a data representation of a telephone call to display a graphical representation of said telephone call. 15

37. The system of claim 36 wherein the graphical representation comprises a representation of each segment of the call.

38. The system of claim 36 wherein the graphical representation comprises a representation of the length of time of each segment of the call. 20

39. The system of claim 35 wherein the display software further displays a table comprising data from the data representation.

40. A method for recording information regarding telephone calls comprising one or more telephone call segments, wherein said calls comprise calls wherein at least one participant participates in a plurality of segments, comprising: 25

- (a) receiving audio data regarding one or more telephone call segments and data regarding telephone events associated with said telephone call segments; 30
- (b) storing the received audio data regarding telephone call segments;
- (c) storing the received data regarding telephony events associated with said telephone call segments; 35
- (d) identifying telephone call segments that relate to one telephone call;
- (e) identifying multiple call segments that have the same participant; and 40
- (f) constructing data representations of lifetimes of telephone calls, wherein each data representation of a telephone call is constructed using data regarding tele-

68

phony events associated with the telephone call segments of the telephone call.

41. The method of claim 40 wherein a data representation of a telephone call comprises:

- (i) a list of participants in the telephone call;
- (ii) a list of telephony events regarding the call;
- (iii) a list containing the time each telephony event occurred; and
- (iv) the start and end time of the call.

42. The method of claim 40 wherein a data representation of a telephone call comprises, for each segment of the call, the location of the stored audio data of that segment.

43. The method of claim 40 wherein the received audio data and the data regarding telephony events is stored in the same memory.

44. The method of claim 40 wherein a data representation of a telephone call is constructed by a plurality of physically separated processors.

45. The method of claim 42 wherein a location of stored audio data of each segment comprises the location of a .WAV file containing the audio data.

46. The method of claim 45 wherein a data representation of a telephone call further comprises an offset within the .WAV file to the start of the stored audio data.

47. The method of claim 40 wherein data regarding telephony events is received from a plurality of sources connected to a telephone switching environment.

48. The method of claim 40 further comprising the step of using a data representation of a telephone call to display a graphical representation of said telephone call. 30

49. The method of claim 41 further comprising the step of using a data representation of a telephone call to display a graphical representation of said telephone call.

50. The method of claim 49 wherein the graphical representation comprises a representation of each segment of the call. 35

51. The method of claim 49 wherein the graphical representation comprises a representation of the length of time of each segment of the call.

52. The method of claim 48 further comprising the step of displaying a table comprising data from the data representation. 40

* * * * *

EXHIBIT F



US006775372B1

(12) **United States Patent**
Henits

(10) **Patent No.:** **US 6,775,372 B1**
(45) **Date of Patent:** **Aug. 10, 2004**

(54) **SYSTEM AND METHOD FOR MULTI-STAGE DATA LOGGING**

(75) **Inventor:** John Henits, Bethel, CT (US)

(73) **Assignee:** Dictaphone Corporation, Stratford, CT (US)

(*) **Notice:** Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) **Appl. No.:** 09/324,459

(22) **Filed:** Jun. 2, 1999

(51) **Int. Cl.**⁷ H04M 7/00; G11B 15/18

(52) **U.S. Cl.** 379/219; 379/207.02; 379/51; 379/93.02; 360/69

(58) **Field of Search** 379/207.01-207.16, 379/219, 220.01, 221.01-221.15, 88.04, 67.1, 73, 88.13, 88.16, 88.17, 88.22-88.28, 37, 41, 45, 51, 93.01, 93.02; 360/61, 62, 69, 73.01

(56) **References Cited**

U.S. PATENT DOCUMENTS

3,104,284 A	9/1963	French et al.	
3,369,077 A	2/1968	French et al.	
3,723,667 A	3/1973	Park, Jr. et al.	179/100.1
3,727,203 A	4/1973	Crossman	340/174.1 J
4,016,540 A	4/1977	Hyatt	704/258
4,130,739 A	12/1978	Patten	179/100.1
4,199,820 A	4/1980	Ohtake et al.	365/234
4,260,854 A	4/1981	Kolodny et al.	179/6.09
4,298,954 A	11/1981	Bigelow et al.	364/900
4,360,854 A	11/1982	Schergen et al.	361/149
4,435,832 A	3/1984	Asada et al.	381/34
4,442,485 A	4/1984	Ota et al.	364/200
4,542,427 A	9/1985	Nagai	360/72.1

4,549,047 A	10/1985	Brian et al.	179/18 B
4,573,140 A	2/1986	Szeto	710/52
4,602,331 A	7/1986	Sheth	364/200
4,621,357 A	11/1986	Naiman et al.	370/58
4,630,261 A	12/1986	Irvin	370/81

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

EP	0 554 626 A2	8/1993	H04Q/11/04
EP	0 437 515 B1	1/1998	H04M/3/42
EP	0 822 696 A2	2/1998	H04M/3/36
GB	2 310 102 A	8/1997	H04N/1/21

OTHER PUBLICATIONS

ASC Telecom GmbH, "System DL 2—Digital Voice Logging Unit on DAT—Cassette," Aug. 1992.

Dictaphone Corporation, "Series 9800 (DAT) Digital Logger—Operator's Manual," Nov. 1992.

Press Release, Dictaphone Corporation, "Digital Audio Tape Logger—Another Dictaphone First," Feb. 1992.

(List continued on next page.)

Primary Examiner—Ahmad F. Matar

Assistant Examiner—Benny Q. Tieu

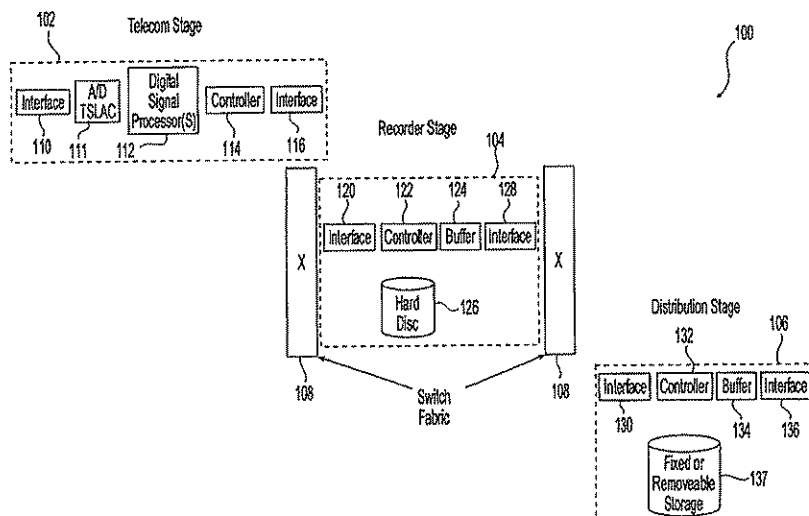
(74) *Attorney, Agent, or Firm*—Jones Day

(57)

ABSTRACT

A multi-stage data logging system comprising a telecommunications stage for receiving, processing, and compressing data from one or more input channels, a recorder stage for storing said data to a memory device, a distribution stage for retrieving said stored data and distributing said data to one or more output channels, and a plurality of interface paths linking said three stages to one another. Different stages of the system can be located wide distances apart and the interface paths linking the three stages can be automatically switched to achieve fault tolerance of the system.

46 Claims, 5 Drawing Sheets



US 6,775,372 B1

Page 2

U.S. PATENT DOCUMENTS

4,631,746 A	12/1986	Bergeron et al.	381/35
4,679,191 A	7/1987	Nelson et al.	370/84
4,686,587 A	8/1987	Hipp et al.	360/74.2
4,688,117 A	8/1987	Dwyer et al.	360/72.3
4,692,819 A	9/1987	Steele	360/72.1
4,709,390 A	11/1987	Atal et al.	381/51
4,785,408 A	11/1988	Britton et al.	364/513.5
4,785,473 A	11/1988	Pfeiffer et al.	379/89
4,799,144 A	1/1989	Parruck et al.	364/200
4,799,217 A	1/1989	Fang	370/68.1
4,811,131 A	3/1989	Sander et al.	360/74.4
4,811,376 A	3/1989	Davis et al.	379/57
4,827,461 A	5/1989	Sander	369/7
4,829,514 A	5/1989	Frimmel, Jr. et al.	370/58
4,835,630 A	5/1989	Freer	360/69
4,841,387 A	6/1989	Rindfuss	360/72.1
4,851,937 A	7/1989	Sander	360/69
4,853,952 A	8/1989	Jachmann et al.	379/88
4,864,620 A	9/1989	Bialick	381/34
4,873,589 A	10/1989	Inazawa et al.	360/53
4,890,325 A	12/1989	Taniguchi et al.	381/34
4,891,835 A	1/1990	Leung et al.	379/88
4,893,197 A	1/1990	Howells et al.	360/8
4,907,225 A	3/1990	Gulick et al.	370/94.1
4,939,595 A	7/1990	Yoshimoto et al.	360/19.1
4,975,941 A	12/1990	Morganstein et al.	379/88
4,991,217 A	2/1991	Garrett et al.	381/43
5,001,703 A	3/1991	Johnson et al.	370/79
5,031,146 A	7/1991	Umina et al.	365/189.01
5,065,428 A	11/1991	Mitchell et al.	380/23
5,127,003 A	6/1992	Doll, Jr. et al.	370/110.1
5,129,036 A	7/1992	Dean et al.	395/2
5,130,975 A	7/1992	Akata	370/60
5,142,527 A *	8/1992	Barbier et al.	370/270
5,163,085 A	11/1992	Sweet et al.	379/89
5,163,132 A	11/1992	DuLac et al.	395/275
5,179,479 A	1/1993	Ahn	360/72.1
5,195,128 A *	3/1993	Knidl	379/88.24
5,199,062 A	3/1993	Von Meister et al.	379/67
5,210,851 A	5/1993	Kato et al.	395/425
5,216,744 A	6/1993	Alleyne et al.	395/2
5,270,877 A	12/1993	Fukushima et al.	360/48
5,274,738 A	12/1993	Daly et al.	395/2
5,305,375 A	4/1994	Sagara et al.	379/89
5,339,203 A	8/1994	Henits et al.	360/39
5,353,168 A	10/1994	Crick	360/5
5,396,371 A	3/1995	Henits et al.	360/5
5,404,455 A	4/1995	Daly et al.	395/275
5,446,603 A	8/1995	Henits et al.	360/48
5,448,420 A	9/1995	Henits et al.	360/48
5,457,782 A	10/1995	Daly et al.	395/2
5,625,890 A	4/1997	Swift	455/67.1
5,813,010 A *	9/1998	Kurano et al.	707/100
5,819,005 A	10/1998	Daly et al.	395/2.09
6,094,673 A *	7/2000	Dilip et al.	709/202

OTHER PUBLICATIONS

Secure Surveillance Systems, Ltd., "S³ DR 1000—16—32 Channel D.A.T. System".

Press Release, Dictaphone Corporation, "Dictaphone's New ProLog™ Digital Communications Recording System Provides Revolutionary Cost Savings/Playback Features," Jun. 1993.

Magnasync Corporation, Product Information and Brochures for Digital Voice Logger, Jul. 1992.

Racal Recorders, Inc., "Rapid Access Voice Logging Recorder," Oct. 1991.

Racal Recorders, Inc., "RAPIDAX Instant Recall Recorder," Jun. 1992.

Racal Recorders, Inc., "RAPIDAX Ranger Digital Tactical Logging System" Jun. 1995.

Racal Recorders, Inc., "RAPIDAX Ranger—Technical Specification" May 1994.

"Rapidax Ranger Sales Manual (Provisional)".

Nice, "Disk-Based Audio Storage/Retrieval Systems—DSN-1000".

"Voice Logging: Comverse Reports Initial Success for Its New Digital Voice Logging System," Edge, vol. 7, No. 209, p. 18, Jul. 1992.

Atis Assmann GmbH, Systemtechnics Division, "Multi-channel Monitoring and Recording—Overview," 1993.

Eyretel, Ltd., "Voice Recording Solutions".

Eyretel, Ltd., "E-500 16 Channel Digital Voice Recorder".

Eyretel, Ltd., "E1000 Digital Voice Recorder" brochure.

Eyretel, Ltd., "The E1000 Digital Voice Recorder" Information Page, www.eyretel.com/E1000.htm, printed Apr. 1997.

Eyretel, Ltd., "An Outline of Eyretel and the E1000 Digital Recorder".

Eyretel, Ltd., Application Bulletin, "Networking Solutions for Eyretel Digital Voice Recorders".

Eyretel Ltd., Application Bulletin, "Buyers Guide to Digital Voice Recorders for Financial Applications".

Nice Systems Ltd., "Nice-Log for Financial Institutions," brochure, 2 pages.

European Patent Application 0 550 273 A2 (corresponding to reference AK listed above), Dec. 31, 1991.

Patent Abstracts of Japan, Application No. 01267499, Oct. 13, 1989.

"Eventide's Digital Voice Logger," Teleconnect, Jun. 1991, p. 42.

"Questions and Answers: The Eventide VR240 Digital Audio-Logger," Eventide Inc., Mar. 27, 1991.

Racal Recorders, *Wordsafe*, Racal Recorders Inc., Pub. No. 3115-2, Dec. 1990.

TEAC Communications Recorders, CR-320/CR-310, TEAC America, Inc., Aug. 1994.

VR240 Manual, Eventide Inc., Jan. 6, 1992.

Magnasync/Comverse, "DVL 1000" brochure 2 pages.

Magnasync Corp., DVL 1000 Digital Voice Logger brochure, 2 pages.

Comverse/Magnasync, Digital Voice Recording DVL 1000 brochure, 1 page.

Comverse Technology Inc., Audio Disk Observer brochure, 3 pages, 1991.

NICE Systems, Inc., "Customer Profile: Legal & General," brochure, 2 pages.

Eyretel, Ltd., "Digital Interfacing," brochure, 2 pages.

Sel-Tronics, Inc., "Series E-1000—The Intelligent Recorder," brochure, 2 pages.

Racal Recorders, Inc., "Racal Adds Remote 'Replay Over LAN' to Wordnet Voice-Logging Recorder," Jun. 1996.

Comverse Information Systems, "Section 1: Ultra 3000 System," 2 pages, Jun. 1995.

Siemens, "Data Voice: Digital Voice and Data Memory for Convenient Information Logging," Jan. 1994.

Nice Systems Ltd., "Company Profile," brochure.

Eyretel, Ltd., "Networking," brochure.

PCT International Preliminary Examination Report for PCT app. No. PCT/US00/15419, mailed Aug. 22, 2002.

* cited by examiner

U.S. Patent

Aug. 10, 2004

Sheet 1 of 5

US 6,775,372 B1

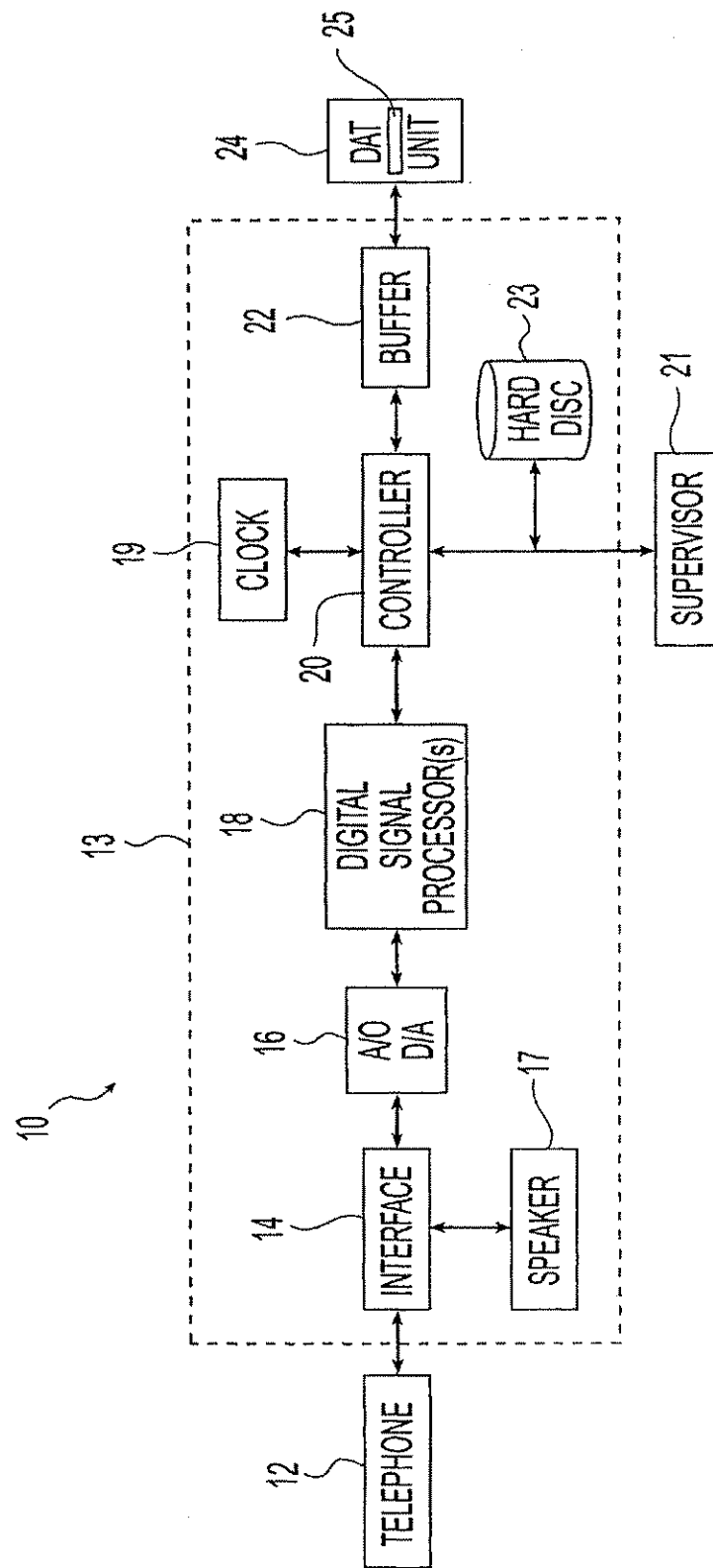


Fig. 1
(Prior Art)

U.S. Patent

Aug. 10, 2004

Sheet 2 of 5

US 6,775,372 B1

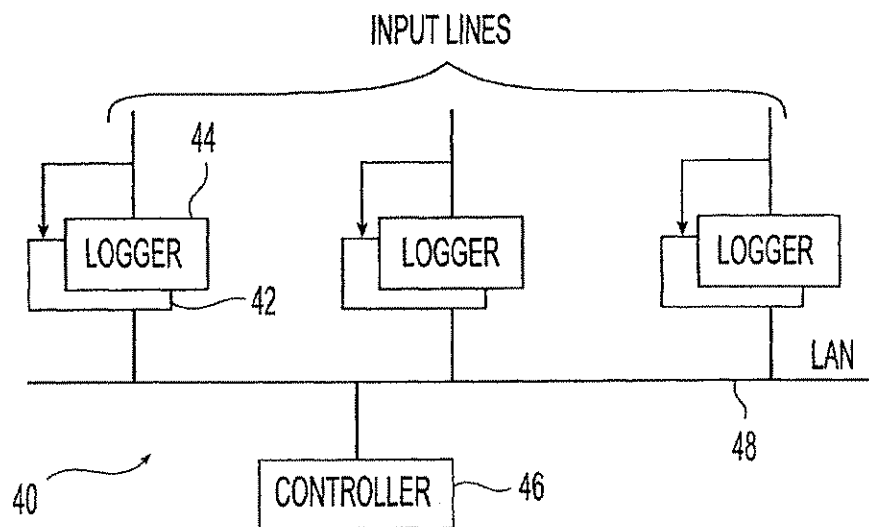


Fig. 2
(Prior Art)

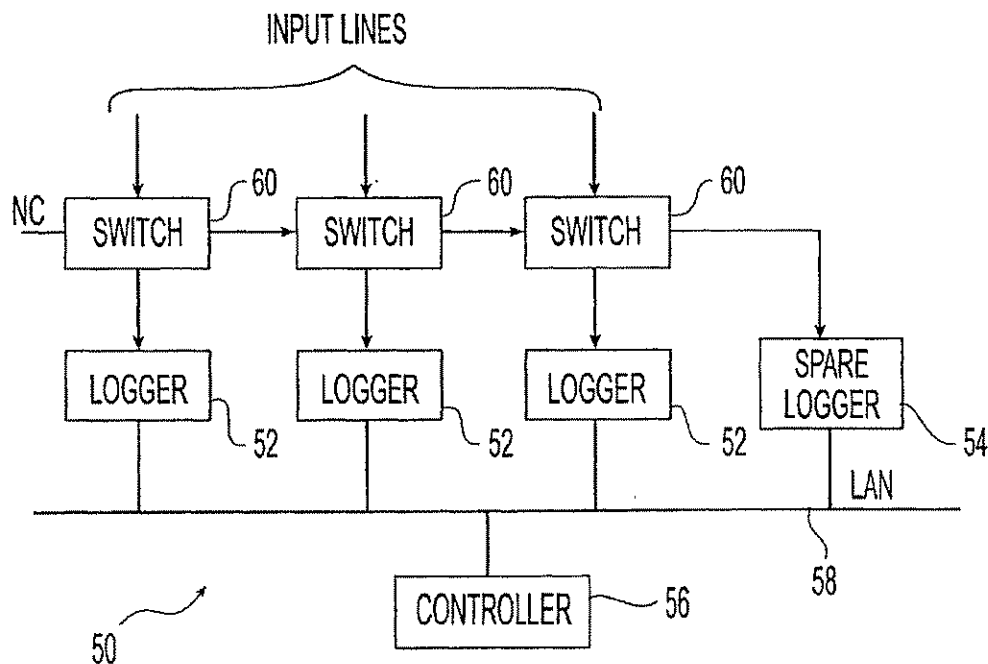


Fig. 3
(Prior Art)

U.S. Patent

Aug. 10, 2004

Sheet 3 of 5

US 6,775,372 B1

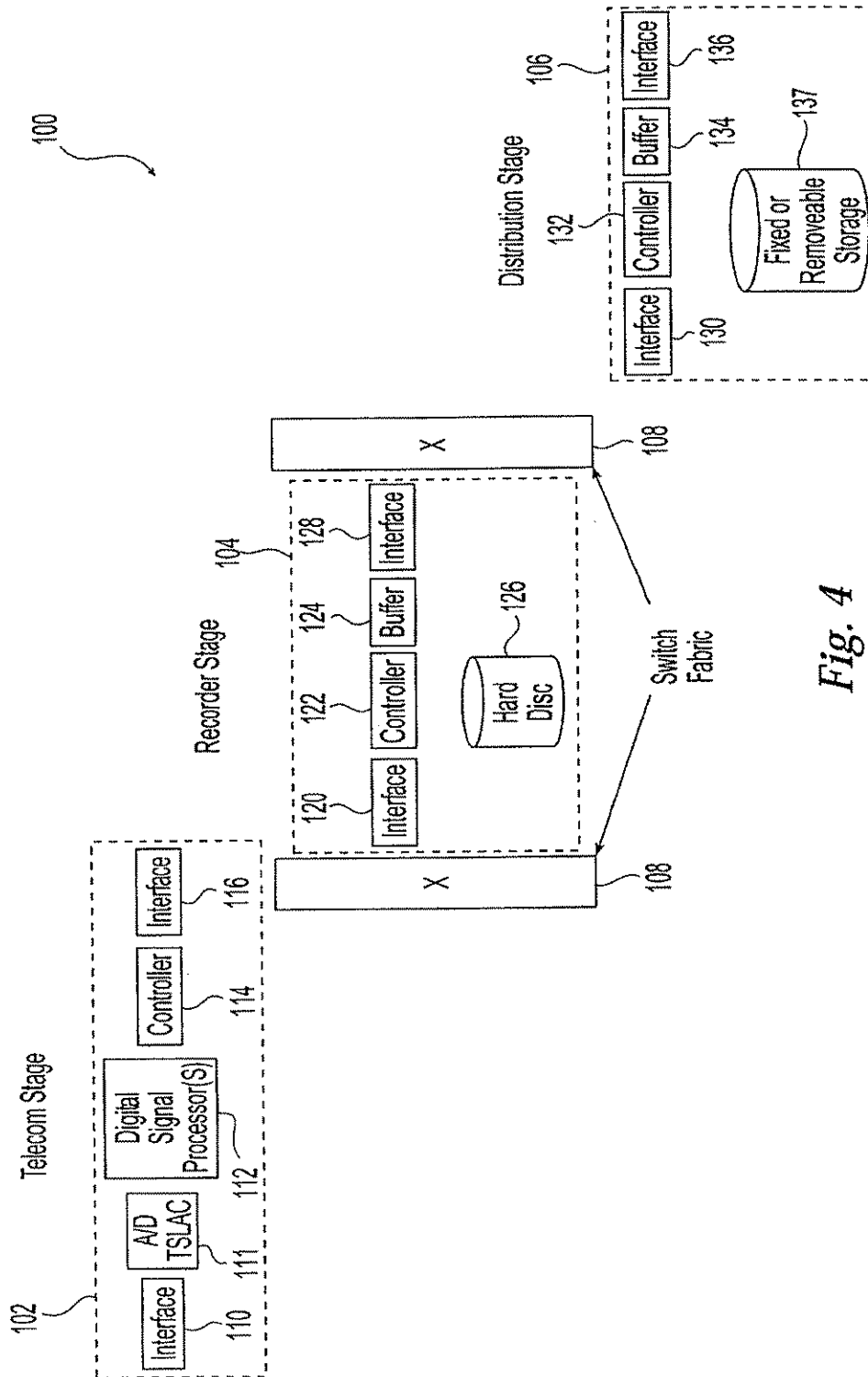


Fig. 4

U.S. Patent

Aug. 10, 2004

Sheet 4 of 5

US 6,775,372 B1

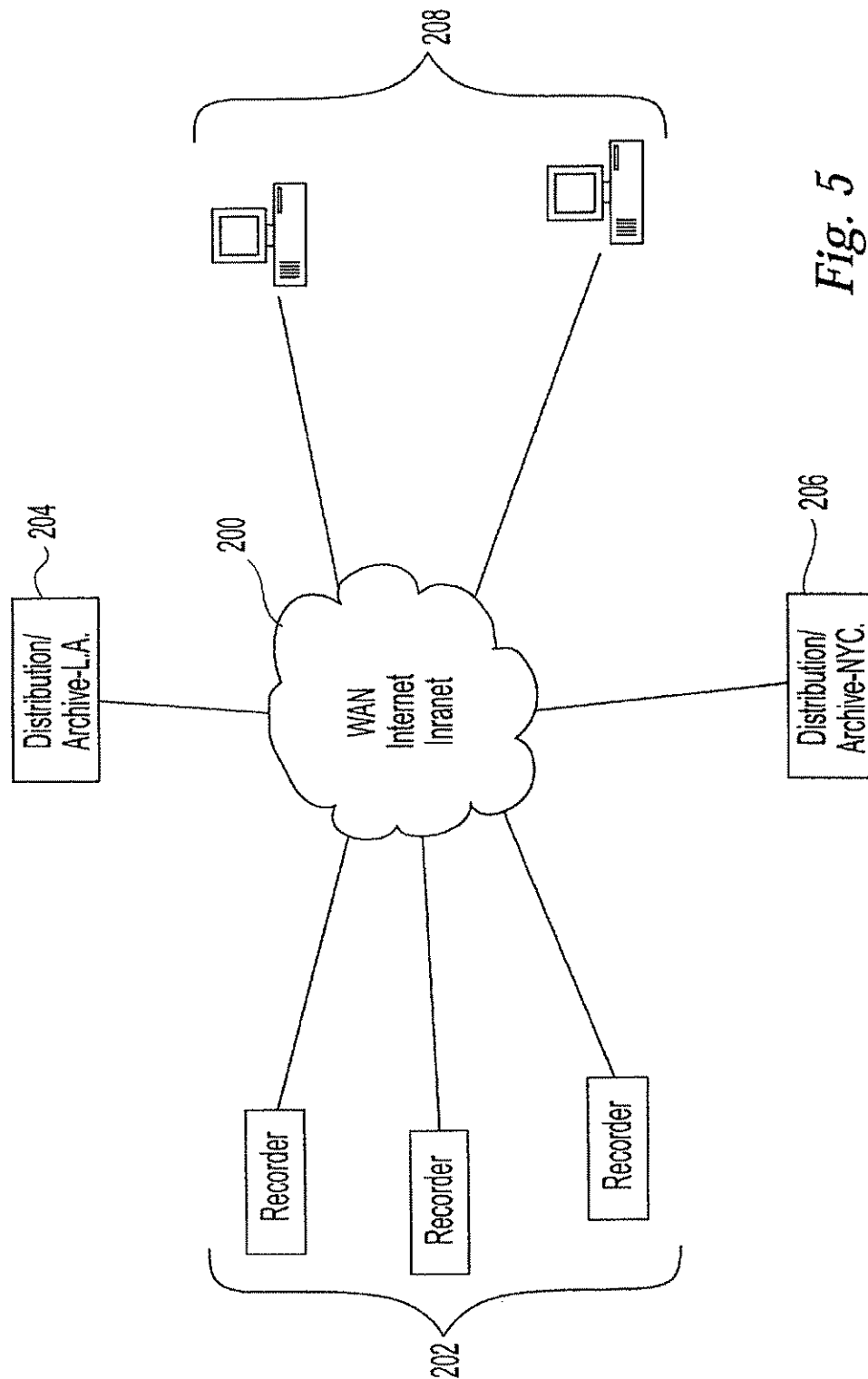


Fig. 5

U.S. Patent

Aug. 10, 2004

Sheet 5 of 5

US 6,775,372 B1

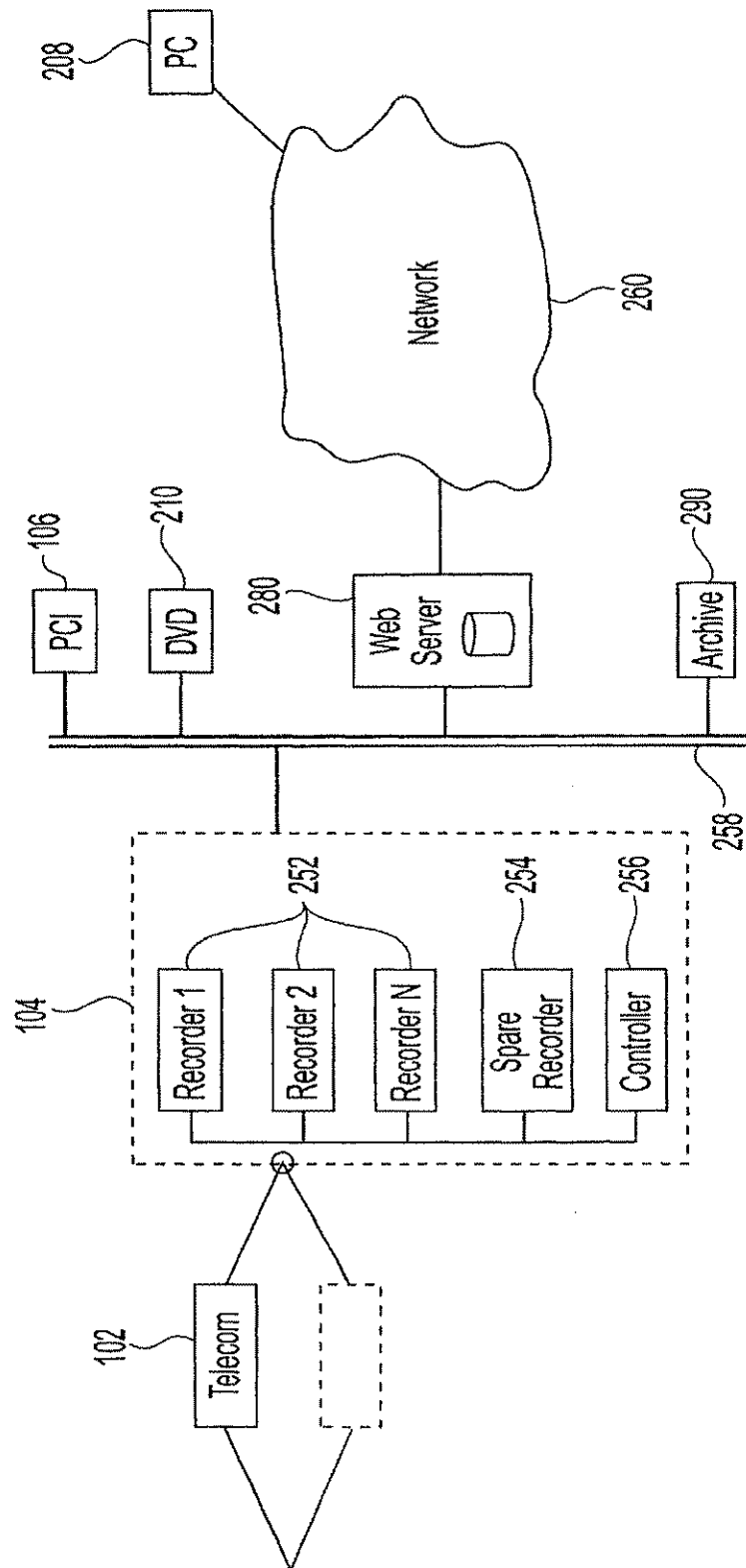


Fig. 6

US 6,775,372 B1

1

SYSTEM AND METHOD FOR MULTI-STAGE DATA LOGGING

FIELD OF THE INVENTION

The present invention relates generally to data logging, and more particularly to a flexible and cost-effective multi-stage system and method for processing multi-channel input data with improved fault tolerance and flexible user access.

BACKGROUND OF THE INVENTION

The efficient storage and retrieval of multi-channel data communications, and especially of voice, are critically important in many modem business and government applications. For example, financial institutions record instructions from clients as a protection against fraud and as evidence in legal proceedings about the content of telephone conversations; public safety agencies record emergency calls for event reconstruction and future investigations; commercial entities monitor transactions over the phone to evaluate salespersons' efficiency, to ensure customer satisfaction and to develop training programs. There is a growing need for reliable recording of multiple channels of multimedia information. These are but a few examples of applications in which it is necessary to efficiently store and process, usually at one location, multiple and frequently simultaneous communications from a large number of incoming data channels. Recent advances in multimedia applications further dictate the necessity to develop practical tools to efficiently process multiple channels in which incoming data can be in different formats, i.e., sound, images, data, etc. It is also apparent that with the proliferation of different communications media, computer platforms and operating systems, a very important practical aspect of all these applications is the ability to provide seamless and efficient interface between the multi-channel data system and its users.

Thus, while capturing the incoming information remains the main function of modem multi-channel data processing systems, other desirable system functions, such as the efficient storage, indexing and retrieval of recorded communications, are becoming increasingly important. Preferably, all system functions should be transparent to the users, regardless of the data format, of the communications media or the specific computer platforms being used. The present invention addresses the need for such a data logging system and method and illustrates their use in practical applications.

Data logging systems for capturing and recording massive volumes of data transmitted over multiple communication lines are known in the art. A typical prior art data logging system, such as the one shown in FIG. 1, is implemented as a single physical unit performing most or all required functions, including telephone interface, signal processing, random access buffering, and archive storage. In many practical applications such implementation is perfectly adequate. Examples of such systems are provided in, for example, U.S. Pat. Nos. 5,819,005; 5,448,420; 5,446,603; 5,396,371; 5,339,203 and 4,827,461 to the assignee of the present application. These patents are hereby incorporated by reference for all purposes.

However, as market needs change and performance demands increase, improved logger architectures have to be designed to support expanding capacity requirements and emerging processing needs. Two strong market demands render prior art logger architectures at a disadvantage in this regard: the need to increase the number of input ports, and the need to provide simple and efficient access to the recorded information in platform and media-independent ways.

2

Turning first to the input port requirements, modem data logging applications dictate the need to support an ever-increasing number of input channels. It should be apparent that as the number of channels increases so does the complexity of the processing system. The increased complexity in turn creates at least two potential problems: (a) diminished fault tolerance of the system; and (b) practical constraints on the physical design, including the weight and size of the system, its wiring and power requirements, among others.

Increasing the system's complexity generally results in a greater vulnerability and higher risk of data losses because of the increased probability that one or more system components can malfunction. However, people of skill in the art would recognize that in many applications it is critically important that the operation of the system remain error-free. Because of this stringent requirement and the fact that the information that loggers are entrusted to record is almost always original and ordinarily cannot be artificially regenerated, it is necessary to build into the system sufficient redundancy so that the malfunction of one or more components would not lead to a shutdown of the entire system. This in turn makes it increasingly important to devise a scalable system that provides fault tolerant characteristics regardless of the number of input channels.

Fault tolerant architectures that require additional "standby" hardware to replace failed components are known in the art. One such architecture, shown in FIG. 2, utilizes a local area network (LAN) 48 for communication between loggers 44 and controller 46. Each logger is responsible for recording communications on a plurality of input lines. In order to provide backup logging capabilities in the event a logger 44 fails, spare loggers 42 are provided. In particular, each logger 44 is associated with a spare logger 42 that is kept ready for use in the event the working logger 44 fails. This fault tolerant architecture is wasteful and impractical, because it requires more space and twice the hardware (at twice the cost) of a comparable standard logger system.

An improved architecture of a fault tolerant system, shown in FIG. 3, is described in an application by Yosef, EP 0822696A2, the content of which is hereby incorporated by reference. The Yosef approach is to use a single spare logger to back up a group of N loggers, thereby reducing the hardware cost in the case when $N \geq 2$ compared with the system shown in FIG. 2. Notably, however, even in this improved prior art architecture the spare unit is a completely functional device capable of performing all logging functions. Accordingly, it does not take advantage of the fact that in operation certain components of the logger fail much more frequently than others. This is the case, for example, in components with physically moving parts, such as hard drives. In accordance with the present invention cost savings are realized for the same fault tolerance level using separation of the logger into two or more functional stages, where higher redundancy is provided for the components having higher failure rates. As illustrated below, using this approach it is possible to reduce the cost of the system considerably, without affecting its performance level.

As mentioned above, the increased complexity of the system also poses wiring, size and other practical problems. For example, an important consideration when installing a logger with a high number of input channels is the number of wires that interconnect the logger to the PBX. Thus, a logger having 128 input channels requires at least 256 wires. For practical reasons including susceptibility to picking up audible electrical noise over long cable runs, it is desirable to install the logger next to or near the PBX. However, this location may not always be optimal, desirable or even possible in practice. In particular, spatial constraints as well as security issues pose serious problems. Thus, the large size

US 6,775,372 B1

3

of a high-capacity logger can make it impossible to install it in certain locations. It may further be costly and/or impractical to run massive cable bundles from the PBX room to the loggers. For security reasons it may not even be desirable that the logger is anywhere near the PBX. It should thus be apparent that prior art systems implemented as single physical units suffer from some very serious practical problems. In accordance with the present invention, separation of the logger into several functional stages provides the flexibility to perform different functions at spatially different locations, i.e., in different parts of the same building, city, or the world. In view of the above, the potential advantages of this approach should readily be apparent.

In addition to the requirements for increased capacity, a whole new set of problems is created by the market demand for flexible playback or data access to the stored information. In the context of this application, access to the recorded data is referred to as data "distribution". People of skill in the art will appreciate that it is by no means a trivial task to design a system capable of recording and processing multiple channels of information, while at the same time providing concurrent access to the recorded information from a large number of users. The task becomes even more complex if the users employ different computer platforms.

While many attempts have been made in the past to address these and other problems associated with prior art data logging, no adequate solution has been proposed so far. The system and method of this invention overcome problems associated with the prior art and thus are believed to present significant technological advance. In particular, separation of the logger into several functional stages provides considerable flexibility in designing a functionally and ergonomically optimized system for use in various practical applications. Further, distributing logger functions into separate physical stages leads to significant and unobvious advantages over the existing practice of utilizing a single physical unit. For example, hardware for performing certain functions in the same system can be bulky and thus inconvenient for positioning at a particular location. Due to the functional separation approach of the present invention, however, bulky parts can advantageously be placed where appropriate. Thus, the relatively bulky recording equipment capturing transaction information in the New York Stock Exchange can be placed miles away from the cramped floor of the Exchange. Additionally, in accordance with the principles of the present invention the design of the distribution stage enables users to access the recorded data at their convenience either real-time or at a later point. Importantly, fault tolerance can be provided at a fraction of the cost associated with prior art approaches. The proposed approach provides better system scalability, reliable performance at a low cost and great flexibility in the retrieval of stored information compared with existing designs.

SUMMARY OF THE INVENTION

Accordingly, it is an object of the present invention to provide a multi-stage data logging system that overcomes problems associated with prior art solutions, and helps meet the market demands for increased input capacity, high distribution flexibility and fault tolerance.

In a preferred embodiment, the invention is a multi-stage data logging system comprising: a telecom stage receiving input from a plurality of input channels; a recorder stage having one or more recorders, at least one recorder storing data associated with input received from at least one of said plurality of input channels; a distribution stage providing access to data stored in the recorder stage; a first interface linking the telecom and the recorder stages and a second interface linking the recorder and the distribution stages; wherein at least two stages of the system are physically separable and in operation can be located wide distances apart.

4

In a specific embodiment the telecom stage of the system further comprises a first interface capturing signals from said plurality of input channels; one or more signal processors converting captured signals into data having a predetermined format; and a second interface for transmitting said converted data to said recorder stage. In embodiments where the input signal can be analog, the telecom stage of the data logging system further comprises at least one analog to digital signal converter.

In specific embodiments, the data logging system of the present invention provides data compression; time stamping of the received input; authentication of signals from the input channels; and encryption of the converted data.

In a preferred embodiment the data logging system of the present invention comprises an archive storage device, part of the recorder and/or telecom stages, for archiving data. In specific embodiments this device can be fixed, such as a RAID array, or removable.

In another preferred embodiment, the recorder stage of the data logging system comprises at least one backup recorder, and the system has means for detecting malfunctions in recorders of the recorder stage, and for automatically switching interface links from the detected malfunctioning recorder to the backup recorders. Related to this embodiment is another aspect of this invention, which is a method for operating a multi-stage data logging system comprising: detecting a malfunctioning recorder in the recorder stage; automatically switching interface links from the detected malfunctioning recorder to a backup recorder to ensure uninterrupted operation of the system; and without disrupting the operation of the system replacing the detected malfunctioning recorder with a functioning recorder.

In yet another aspect, the invention is a method for increasing the capacity of a multi-stage logger system comprising: without disrupting the operation of the system attaching to a network-based or four-wire-based interface between the telecom and the recorder stages at least one recorder so that the combined capacity of the recorders in the recorder stage is equal to or exceeds a given number of channels; and/or attaching at least one additional telecom block to increase the input channel capacity of the system.

In another preferred embodiment, the present invention is a data logger, comprising: a telecommunication device receiving input from a plurality of data sources; a processor converting input from said plurality of data sources to one or more data formats; a memory for storing converted data corresponding to the received input from said plurality of data sources; a communication path; and a server transferring stored data from one or more of said plurality of data sources via the communication path to at least one remote user. In specific embodiments the server is a Web server and the communication path is the Internet.

In another aspect, the present invention is a method for accessing information in at least one digital logger storing data associated with input from a plurality of input channels, comprising: at a Web server having access to said at least one digital logger, receiving a request for retrieval of stored data from a client; retrieving stored data in accordance with the received request; and transferring the retrieved data to the client. In the specific embodiment covering voice input channels, the method further comprises accessing a record of an input channel made by a digital logger; or accessing call information for a record of an input channel made by a digital logger.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be understood and appreciated more fully from the following detailed description, taken in conjunction with the drawings in which:

US 6,775,372 B1

5

FIG. 1 is a schematic diagram of a data logger of the prior art incorporating multiple logging functions in a single unit.

FIG. 2 is a schematic diagram of a logging system of the prior art, employing full operating redundancy where for each working logger there is a spare logger;

FIG. 3 is a schematic diagram of a logging system of a prior art employing one spare logger for each group of "N" working loggers.

FIG. 4 is a schematic diagram of a multi-stage data logging system in accordance with a preferred embodiment of the present invention.

FIG. 5 is a block diagram of a multi-stage logging system configured in accordance with a preferred embodiment of the present invention.

FIG. 6 is a schematic diagram illustrating another preferred embodiment of the multi-stage logger in accordance with the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Prior Art

FIG. 1 is a schematic diagram of a prior art data logger 10. Logger 10 incorporates an interface 14 for receiving data from input lines 12, an analog/digital (A/D) and a digital/analog (D/A) signal converter 16 that converts analog signals from input lines 12 into digital signals when data is flowing in one direction and digital to analog when the signal is flowing in the opposite direction. Interface 14 also communicates with a speaker 17. The system further comprises a digital signal processor 18, generally used to compress the digital signals from converter 16; a controller 20, a buffer 22, and hard disc 23. In a typical implementation all components are located in a single physical unit 13. As shown in FIG. 1, the unit may also include a storage unit 24, such as a digital audio tape (DAT). The configuration and operation of a logger of the type illustrated in FIG. 1 is described in detail in several U.S. patents, including U.S. Pat. Nos. 5,819,005; 5,448,420; 5,446,603; 5,396,371; 5,339,203 and 4,827,461 to the assignee of the present application, and which are incorporated herein by reference.

It will be appreciated by people of ordinary skill in the art that the number of input channels that can be handled by a logger of the type shown in FIG. 1 is usually fixed by the design constraints of the physical unit, i.e., by the number and the capacity of the interface cards, the configuration of the processor 18, and others. Accordingly, to process a higher number of input channels it is generally necessary to either increase the number of interface cards (limited by the number of available slots) or duplicate the logger 10 one or more times, such that each unit handles a particular group of incoming channels. In the configuration shown in FIG. 1, as well as multi-logger architectures in which logger 10 is replicated to increase input channel capacity, users may encounter various problems of the type discussed in the background part of this application. For example, failure of one block in logger 10 may disable the entire unit and thus jeopardizes the recording of several input channels. Thus, the fault tolerance of the system depends on its weakest link, i.e., on the most failure-prone component of the unit. In many cases where no backup unit is provided the resulting fault tolerance is unacceptable. Further, space constraints may pose problems because the standard design of the logger 10 as one physical unit does not allow its separation into functional blocks that can be placed in spatially optimal locations.

Reference is now made to FIG. 2, which is a schematic diagram of a prior art fault tolerant data logging system. Data logging system 40 illustrated in FIG. 2 employs one spare logger 42 for each working logger 44. In a particular embodiment, each logger can be implemented as illustrated

6

in FIG. 1 but other embodiments are possible, as known in the art. In this architecture, all loggers are connected via a local area network (LAN) 48 and upon failure of a working logger 44, the spare logger 42 assigned to the failed logger is automatically activated. While just about any fault tolerant system will require some additional "standby" hardware to replace failed components, system 40 illustrated in FIG. 2 requires twice the hardware, at twice the cost, of a conventional system and in most cases is commercially impractical.

Reference is next made to FIG. 3, which is a schematic diagram of an improved fault tolerant data logging system of the prior art. System 50 illustrated in FIG. 3 consists of one or more groups of "N" loggers 52 connected via LAN 58, and linked by a group of "N" switches 60 to a single spare logger 54. When one logger 52 fails, the switches 60 automatically reroute data from the failed logger's input lines to the spare logger 54. For $N \geq 2$ this system requires less "standby" hardware than the prior art system of FIG. 2. Nevertheless, it is apparent that such a design is suboptimal, at least because it does not make use of the fact that certain components of the logger are more reliable than others. Importantly, this and other prior art logger designs represent a single-unit design philosophy that carries many performance limitations as noted above.

The Present Invention

In contrast to the prior art designs described above, the approach taken in accordance with the present invention is to combine low-cost and high-reliability multi-stage recorders storing data from multiple input data channels with a powerful distribution technology that enables platform-independent access to the stored data from different physical locations. Multiple channel data recording, and particularly voice loggers, are known in the art. As shown above, some steps have already been taken to improve the reliability of such devices using redundancy approaches. From the distribution perspective in the last few years Internet technology is becoming standard. However, to the best of this inventor's knowledge, there is no suggestion in the past of combining these two paradigms in a multi-stage distributed logger, a next-generation product that brings together the most desirable features of these technologies.

In particular, loggers have been used in the past to record, index and store large volumes of data from different input channels. However, practical limitations, such as their single-unit design, have limited the use of such loggers in many applications. With the distributed multi-stage architecture of the present invention, at least the following advantages over the prior art can be expected:

- (a) highly reliable and low-cost data recording and storage due to a novel fault tolerant architecture in which failure-prone components are duplicated for redundancy;
- (b) optimized use of space made possible by the physical separation of the logger in separate stages that allow, for example, placing bulky components away from locations where their presence can be inconvenient (due to space limitations), or even highly undesirable (as in covert operations);
- (c) improved business models for information distribution.

Reference is now made to FIG. 4, which is a schematic diagram of a multi-stage data logging system in accordance with a specific embodiment of the present invention. In the illustrated embodiment, system 100 consists of a telecommunications stage ("telecom stage") 102, recorder stage 104, and distribution stage 106. Broadly, the telecom stage serves to capture incoming signals, convert the captured signals in a predefined format, preferably digital, and perform certain processing on the converted signals, usually in the nature of compression. Recorder stage 104 stores the processed input

US 6,775,372 B1

7

signals to a hard drive or similar large-capacity storage device, as well as in a buffer from which the data can be forwarded to fixed or removable storage devices, which in the embodiment shown in FIG. 4 are part of the distribution stage. In accordance with the present invention, the distribution stage serves not only for access to the stored information but also for archiving purposes. Importantly, individual functional stages of the system 100, such as those illustrated in FIG. 4, can be physically separated and typically are housed in separate physical units.

It should be understood that FIG. 4 is merely an illustration of a system built in accordance with the principles of the present invention and should not be construed as a limitation of its design approach. Thus, functional separation of the logger into more than three stages can be desirable in certain applications. In alternate embodiments functional stages can be combined—for example, with reference to FIG. 4, the telecom and recorder stages on one hand, or the recorder and distribution stages on the other hand can be implemented as single physical units. Furthermore, in alternative preferred embodiments data from the recorder stage is made available to users via a communications network such as private communication lines, corporate intranets, the Internet or any combination thereof. It will be appreciated that such networks can be accessed by various different computers, so that stage 106 is “distributed” and would have no identifiable physical location.

In another aspect of this invention the stages of the system shown in FIG. 4 are connected using switch fabric 108 interfacing the telecom and recorder stages on one hand, and the recorder stage and distribution stages of the system 100 on the other. It will be appreciated that switch fabric 108 interfacing different stages can be different in design and implementation. In accordance with the present invention, the selection and design of the switch fabric 108 linking the individual stages of the overall system 100 are design parameters that allow for the physical separation of the individual stages in accordance with the preferred embodiments. In particular, dependent on the application, the stages of system 100 can be physically separated so that, for example, the telecom and recorder stages can be located in different parts of the same room, in different rooms of the same building, in different buildings of the same town, in different towns or even different continents. In a specific embodiment such wide spatial separation can be implemented using, for example, standard public switched network access arrangements, dedicated lines, or other communications media, as known in the art. The individual stages of the system in accordance with the present invention are considered in more detail next.

The Telecom Stage

In accordance with the present invention telecom stage 102 generally functions to capture and pre-process signals from a plurality of communications lines into a format that is recognized by the recorder stage. As indicated above, in a preferred embodiment the communication lines could be standard telephone lines, dedicated communication lines or other input data sources, which may transmit analog or digital signals. Regardless of their physical source or data format, in a preferred embodiment the captured input signals are digitized and compressed, as known in the art. Naturally, different types of input data can be transformed into different internal formats, as instructed by the user.

More specifically, as shown in FIG. 4, telecom stage 102 incorporates a first interface 110 for receiving signals from the input lines (not shown), which typically are telephone lines. In accordance with a preferred embodiment, interface 110 is designed for analog or digital input signals, or some combination thereof. When used with telephone lines, interface 110 generally is a high impedance interface that allows for passive tapping of the phone lines. In this embodiment

8

interface 110 employs voltage and/or current sensing to trigger recording of corresponding input channels. External triggering signals can also be used at the telecom stage, as known in the art. In the case of a digital input, interface 110 serves as a digital phone coupler, as known in the art. Various alternate embodiments of the interface 110 are known in the art and are described in detail, for example, in U.S. Pat. No. 4,827,461 to Sander, which is incorporated herein by reference. It will be appreciated that interface 110 can be used for various additional functions, such as beep generation, notching and others.

In the embodiment illustrated in FIG. 4, telecom stage 102 further comprises an A/D signal converter 111 for digitizing the received analog signals. Naturally, no such converter is required for purely digital inputs. Stage 102 also comprises a digital signal processing (DSP) unit 112 for processing signals digitized by converter 111 or supplied directly from digital input channels. In a preferred embodiment, DSP unit 112 is capable of handling various input data types, i.e., voice, image, facsimile and others. In a specific embodiment one processor 112 is capable of handling various input data types and its processing algorithm is selected dependent on the input data type. In alternative embodiments, one processor can be used for each input channel so that different processing algorithms can be applied simultaneously to different input channels. It will be appreciated that whether one or more processors are used, different data compression algorithms can be applied to different input channels dependent on the data type, the statistical distribution of the incoming data, and other factors. Such processing algorithms are generally known in the art and will not be considered in further detail.

Telecom stage 102 illustrated in FIG. 4 further comprises controller 114 that operates to direct and monitor the functions of the entire stage. Specifically, in a preferred embodiment, controller 114 directs the application of a particular processing algorithm by DSP processor 112 to a particular input data type. When instructed by the controller, in a specific embodiment processor 112 is capable of performing DTMF signal detection and decoding, and preferably also performs compression of the data in accordance with well established standards. For example, in the case of voice signal recording, the output of DSP processor 112 is preferably made compatible with the G 723.1 and G 722 voice compression standards, which are known in the art.

In a specific embodiment, interface 110 and A/D converter 111 together with a time slot assignment circuit (not shown) are implemented as a logger coupler card, that places the input data on a pulse code modulation (PCM) highway (not shown) within the telecom stage 102. In this case the DSP unit 112 is directed by the controller 114 to process signals in the appropriate time slots. In particular, controller 114 reads the system clock, directs the assignment of time slots to individual input channels, and provides a time stamp that is used in the creation of individual channel records. As noted above, block 114 also controls the operation of the DSP unit 112 and distributes data compressed at its output to interface 116. In a preferred embodiment an important function of the controller is to reconfigure the processing algorithm(s) of the DSP unit 112 using downloaded software. Thus, in a specific embodiment an input port (not shown) is provided for the controller to download instructions used to re-program the DSP processor(s) 112 from a personal computer (PC).

Finally, in a preferred embodiment telecom stage 102 comprises a second interface 116 for transmitting compressed data to the recorder stage 104. In a specific embodiment the second interface 116 may be an E1 (2 Megabit) transceiver or, for smaller applications, an RS-485 (1 Megabit) line. Other embodiments may be used, as known by those of skill in the art.

US 6,775,372 B1

9

In accordance with the present invention, certain functions that are not required but are frequently useful in the operation of data loggers are also provided in specific embodiments. Such functions include, for example, time stamping, encryption and authentication of incoming data. Generally, authentication refers to mechanisms by which the transacting parties prove they are who they claim to be, i.e., in this context that data from an input channel indeed comes from a particular source; encryption usually refers to the altering of data so that it cannot easily be read or modified, if intercepted. If used, such functions should be placed as close as possible to the source of the information that is being captured—and accordingly are implemented in the system of this invention as part of the telecom stage 102. In a specific embodiment, these functions are implemented by the controller block 114. Other functions of the telecom stage can be implemented in specific embodiments, as known in the art. Thus, for example, in a specific embodiment telecom stage 102 can provide lightning protection and can be used to detect DTMF signals, to identify caller IDs, to detect the presence or absence of voice in a particular input channel, and correspondingly control a trigger mechanism for use by the recorder stage 104. As known in the art, the telecom stage can also be used as a current and/or voltage sensor. Functions described in this paragraph can be implemented as part of the logger coupler card used in a specific embodiment, and/or the DSP unit 112 of the telecom stage 102.

It will be appreciated by people of ordinary skill in the art that in general the components of the telecom stage are robust and in particular involve no physically moving parts. Accordingly, in a preferred embodiment, they need not be duplicated in fault-tolerant architectures built using the principles of the present invention. Naturally, in very sensitive applications where there is little or no margin of error the telecom stage can also be duplicated with a standby unit, as shown in FIG. 6. However, in accordance with the present invention, a single standby telecom stage can be used to serve as backup of N recorder stages in a manner similar to that illustrated in FIGS. 3 and 6. The cost savings compared with the architecture shown in FIGS. 2 and 3 are readily apparent.

Finally, since telecom stage 102 is separated from the remaining components of the logger by switch fabric 108, the stage can be implemented in a separate physical unit having as many input channel slots as desired. This compares favorably to the standard single-unit logger design, in which the number of input channels is limited by the number of available slots on the circuit board, which in turn are limited by the overall dimensions of the unit. It will be appreciated that the multi-stage design used in accordance with the present invention thus extends the input channel capacity of the logger.

The Recorder Stage

Again with reference to FIG. 4, recorder stage 104 of the logger stores input from the telecom stage to a hard drive or a suitable large-capacity random access storage device, as well as in a buffer from which the data can be forwarded to fixed or removable storage devices. In a particular embodiment directed to storing voice records, the function of recorder stage 104 can be described broadly as creating voice files and providing an associated database with stored call information. To this end, the recorder stage 104 in a preferred embodiment has the ability to create new voice records, merge call records and establish a database with call information for each created record. In a preferred embodiment, newly created records, along with identifier information are stored in a database at unique record addresses.

In accordance with the embodiment illustrated in FIG. 4, recorder stage 104 has the following components: (a) an

10

interface 120 receiving input from the telecom stage; (b) a controller 122 directing and monitoring the recorder stage operations, and generally comprising a microprocessor with memory; (c) an elastic buffer 124 for transitional data storage; (d) a hard disk or similar large-capacity storage device 126 for data storage (2 Gigabytes minimum); and (e) a second interface 128 for connection to the distribution stage. It will be appreciated that while the components of the recorder stage are illustrated in FIG. 4 as being separate, they need not be implemented in separate physical units.

The function and operation of the recorder stage 104 is generally known in the art. For example, it is described in U.S. Pat. Nos. 5,819,005; 5,448,420; 5,446,603; 5,396,371; 5,339,203 and 4,827,461 to the assignee of the present application, which are hereby incorporated by reference for all purposes. It will be appreciated that alternate embodiments employing different storage media or internal architecture can be used, if necessary. For the purposes of this invention, it is important to note two main differences from prior art designs. The first difference is due to the flexible platform-independent data access provided in accordance with the present invention to end users in the distribution stage. To support such an access, when viewed from the distribution stage the recorder of this invention appears as a file server providing access to a plurality of records (from input data sources) stored in a database at unique record addresses. The server preferably supports different data transfer protocols. The second difference is that since the recorder stage uses moving physical parts and is thus prone to malfunctioning, for high reliability in accordance with this invention it is desirable to provide backup units of the N recorders (or critical components thereof) of the recorder stage. These two aspects of the present invention are described in further detail next.

In accordance with a preferred embodiment, recorder stage 104 has operating system that supports different file access protocols, such as Microsoft's SMB, the UNIX-based NFS and the standard FTP file transfer protocols. Thus, when viewed from the distribution stage 106, the recorder in a preferred embodiment appears as a server supporting these and possibly other file transfer protocols, in which records that correspond to different input channels appear to the user as separate files having unique record addresses. In a preferred embodiment, these files are identified by the call record information and contain, for example, information about the caller ID, the date and time of the communication, its duration, and others. With reference to FIG. 6, a user at a PC 106 connected to the recorder stage via communications path 258 and operating, for example, in a Microsoft Windows environment can click on a "Network Neighborhood" icon on his computer to locate a recorder 202 of interest. As noted above, various records in such a recorder would appear as files that are viewable using Microsoft's SMB protocol. Once a desired record is identified, it can be "dragged" to the user's computer for playback. Similarly, a NFS or Linux client can access files at a recorder that supports the corresponding UNIX-based protocol. Further, file transfers from the recorder can be accomplished over a network 260, such as the Internet, using the well-known FTP protocol. Naturally, security can be implemented at the recorder end, if desired, as known in the art. In each case, it is important that from the viewpoint of the user the recorder stage appears as a server accessing different files organized in such a manner as to provide the user with access to unique records from different input channels.

Focusing next on the fault-tolerant aspect of this invention, FIG. 4 illustrates a recorder stage with a single recorder. As noted above, however, in many practical applications the recorder stage can have two or more recorders, each processing information about a particular set of input channels. Furthermore, in terms of the fault-tolerant aspect

US 6,775,372 B1

11

of the present invention and with reference to FIGS. 2, 3 and 6, it is noted that unlike the system's telecom stage, the recorder stage typically comprises physically moving parts, and thus is prone to occasional malfunction. Accordingly, in a specific embodiment of the present invention shown in a diagram form in FIG. 6, the recorder stage is implemented in a single physical unit in what is known as a "hot swapping" M×N design. In this design one backup recorder unit 254 is used for every N working units 252, and switchover to the backup unit occurs automatically without turning off the power. In a specific embodiment, N recorders implemented as swappable modules are physically placed in drive bays of a physical unit and can be replaced, if necessary, without turning off the power of the unit. Detection of a recorder malfunction and control over the entire stage is provided by the controller 256, as known in the art. Schematically, the architecture of the fault-tolerant embodiment of the present invention resembles the architecture in FIG. 3, the important difference being that instead of using an entire backup logger, the system of the present invention uses as duplicate hardware merely the recorder stage or components thereof, such as the hard disk 106. Such a design approach not only provides the desired performance level and fault tolerance but also results in considerable savings in terms of the overall system cost.

Providing automatic switchover without disrupting the operation of the system, as described, is an important aspect of the present invention. This feature is made possible by the selection, in a preferred embodiment, of network or 4-wire interfaces linking individual stages of the logger system, which enables inserting or taking out system components without affecting the operation of the entire system. It will be appreciated that the same feature would be difficult to provide with PC bus interfaces. Further, it will be appreciated that this selection of the interface links also makes it possible to incrementally add recording capacity simply by attaching to these links of additional telecom blocks, recorders or both. Clearly, this incremental addition can also be done without interrupting the operation of the system, which feature of the present invention is perceived to have significant practical utility.

The Distribution Stage and Switch Fabrics

In accordance with the present invention a very flexible approach is taken to the design of the distribution stage of the logger. Generally, the distribution stage serves for retrieval of recorded information and providing it in a humanly recognizable form, i.e., as an image, a printout, a sound clip or others. In a preferred embodiment, the distribution stage also serves for archiving the recorded information to a removable storage, such as magnetic tape, magneto-optical storage device, DVD, or others.

As best seen in FIG. 6, in a preferred embodiment the distribution stage may have no single physical location, and in this sense is truly "distributed". As seen in this figure, one or more recorders 252 are attached to a communication path 258. This path can be provided using Ethernet, optical fiber or other media. It may be part of a local area network (LAN), a wide area network (WAN), and preferably has access to the Internet. In a specific embodiment, the communication path is implemented as a Universal Serial Bus (USB). It is foreseen that the communication path may also be part of a storage architected network (SAN). As seen from the above list of options, there is no limitation on the type of communication path provided at the back end of the recorder stage, which path concurrently serves as part of switch fabric 108 (see FIG. 4).

With reference to the specific embodiment illustrated in FIG. 4, the distribution stage also has a controller 132 for directing and monitoring distribution stage operations, a buffer 134 for transitional data storage, and a second interface 136 for distributing data to one or more output chan-

12

nels. Distribution stage 106 in the embodiment illustrated in FIG. 4 may also include an archive storage device 137 for archiving data, which can be either fixed (such as a RAID array) or removable. In one embodiment, this archive storage device 137 is a DVD RAM drive, a digital audio tape (DAT), or any suitable device capable of storing large volumes of data from multiple recorders of the recorder stage 104. In an alternate embodiment, the archive drive is part of the recorder stage, not the distribution stage. As seen, in this "OEM-friendly" embodiment, users may be free to design their own customized distribution stage.

With reference to FIG. 6, in a preferred embodiment, the distribution stage is implemented using at least in part Web server design technology. This approach has the advantage of using a powerful distribution tool, which is well known and can readily be employed to support a number of users regardless of their specific computer platforms. In this sense, the Web server 280 acts as an intermediary between one or more recorders 252 in the recorder stage of the logger, and the users accessing the stored information via, for example, the Internet. As noted in the preceding section, in accordance with the present invention the recorder stage will make records from different input channels available as files that are stored at unique record addresses and are made accessible to the Web server 280. In this context it should be noted that in a preferred embodiment the Web server used in the present invention has built-in archiving functions.

In accordance with the present invention Web server 280 may be implemented using any computer, such as, for example, a SUN work station using the UNIX operating system and running a web server program that preferably accepts requests for information framed according to a suitable protocol, such as the HyperText Transport Protocol (HTTP) or a version of it that supports public-key-based authentication or encryption. In response to these requests, Web server 280 accesses the loggers records directly, or it causes a process to access data in a database of the recorder stage through a common gateway interface (CGI); and sends the requested files to the requesting client according to the client's Internet address which, in one embodiment, may be provided according to the Transmission Control Protocol, Internet Protocol (TCP/IP). In a specific embodiment, access to the server can be provided after going through a firewall (not shown) for added security. It will be appreciated that similar distribution scheme can be provided as part of a corporate intranet.

In a specific embodiment employing a Web server as part of the distribution stage users 208 access the Web server through a browser or another suitable application that can playback audio files, in the case of recorded voice information, or display data in a generic case. Specifics of the file formats or data transmission protocols are generally known in the art and need not be described in further detail.

Reference is now made to FIG. 5, which is a block diagram of a multi-stage logging system configured in accordance with a preferred embodiment of the present invention. A wide area network (WAN) 200, such as the Internet or alternatively an Intranet, links a series of recorders 202 located in various different cities, to distribution/archiving engines 204 and 206, located in principal cities like Los Angeles and New York, respectively. A series of user computers 208, possibly located in different cities, is also connected to WAN 200 and can access data recorded by recorders 202 and stored in archive drives 204 and 206. In this configuration, agents located in offices, or even their own homes, in different cities throughout the world can access data logged by equipment physically located thousands of miles away.

As noted above, with reference to FIG. 4 each stage of the logger in accordance with the present invention is designed so that the switch fabric units 108 can be used not only to

US 6,775,372 B1

13

provide physical separation of the individual stages of the logger, but also to achieve fault tolerance of the system by isolating failed components and switching operation to properly functioning standby units. Numerous configurations are possible, as illustrated in exemplary embodiments above.

Low cost components and switch fabric design also allow for fault tolerant capability in a scalable fashion. To demonstrate the fault tolerant characteristics of a typical configuration, consider a recorder stage comprised of eight working recorders and three "spare" recorders. Assuming a 2 Megabit (4 wire) interface linking the recorder stage with the telecom and distribution stages, the eight recorders would be linked to the three spares via a 32x12 switch matrix ((8 working recordersx4 wires)x(3 spare recordersx4 wires)). In the event of a recorder failure, the recorder controller would route the failed recorder's input to a spare recorder via the switch matrix. The cost savings compared with the case of duplicating entire units are readily apparent.

The advantages of the data loggers built in accordance with the present invention are illustrated in the following examples:

EXAMPLE 1

One of the advantages of the multi-stage distributed logger of the present invention—its design flexibility—is clearly illustrated by its use in stock or commodities exchanges. Financial institutions represented at such exchanges record instructions from clients as a protection against fraud and as evidence in legal proceedings about the content of telephone conversations. Therefore, it is highly desirable that data loggers be provided to record communications between brokers at the exchange floors and their clients. As known, however, there is always a concern about the space on the trading floors. This concern is specifically addressed by the multi-stage design of the logger in accordance with the present invention, in which only the telecom stage electronics would need to be located physically on the trading floor. The recorder stage and various distribution servers of the logger could be located physically either someplace more convenient in the building, or in fact someplace else in the city or the world. For example, an E1 twisted pair interface can stretch approximately 2000 feet, so with available "repeaters" the separation distance between the telecom and the remaining stages is virtually unlimited. Since the voice compression function is performed in a preferred embodiment in the telecom stage, use of the public switched network (or private) facilities would be optimized. Importantly, the telecom stage can apply encryption algorithms so that data coming out of the exchange floor is protected against unscrupulous use. It will be appreciated that the same advantage can be used, for example, by law enforcement agencies that are entitled to have wiretap access to phone lines, but until now did not have a mechanism to protect themselves by encryption from being wiretapped themselves.

Furthermore, it can be appreciated that the multi-stage design of the logger in accordance with the present invention enables considerably simplified disaster recovery procedures, since destruction or simply failure of any single component would not affect the function of the entire system. In addition, since the present invention separates the telecommunications electronics from the recorder, cabling to the recorder can easily be reduced. Such cabling reductions may range from 15:1 to 200:1. The net result is lower installation costs and the satisfaction of any political concerns regarding system location and security.

EXAMPLE 2

In this example it is shown how the multi-stage design of the data logger in accordance with the present invention can

14

be used to create and implement new business models. Assume for example that a data logger is used by a police to monitor 911 calls in a particular area. Information about such calls is generally available to the interested public, for example the news media, but so far has been used in only a few cases primarily because of the difficulties associated with the access to such information. Using the distributed design of a logger in accordance with the present invention, however, it would be a simple matter to make such records available to the public immediately, possibly for a fee.

In particular, a Web server run by the police can store or be given access to data files corresponding to individual 911 calls. With reference to FIG. 6, an authorized user may utilize a personal computer to access the police logger over the world wide web (WWW) of the Internet at a predetermined URL. The user's computer may be running standard web browser, such as the NETSCAPE browser. As soon as the computer is connected to Web server 280 a computer process starts communicating with the user through the web browser. In particular, a service routine may be initiated causing a "home page" to be displayed, which greets the user, and describes the service provided by system. Next, Web server 280 elicits user information, including a user identification (ID), password and other administrative data necessary for ensuring that the user is an authorized user. Next, the audio file(s) corresponding to individual 911 call records can be displayed to the user who can then select a file. As known in the art, various protocols exist for the playback of audio files over the Internet. By means of an example, the user can be provided with a File Header information defining the audio files in fields such as: (1) File name; (2) Date created; (3) Size in bytes; (4) Audio file types in use; (5) Total associated files; (6) a command set code; and others. Additionally, instructions may be provided for decompressing and decoding a specific or proprietary Audio Player software (possibly residing on the Web server), on how to play the files, including hyperlinks, etc. Various levels of access protection can be implemented, as known in the art, so that access hierarchy can be established for different groups of users.

It should be apparent that the use of the logger along the lines described in Example 2 creates the possibility of a completely new and heretofore unused business model.

While the foregoing has described and illustrated aspects of various embodiments of the present invention, those skilled in the art will recognize that alternative components and techniques, and/or combinations and permutations of the described components and techniques, can be substituted for, or added to, the embodiments described herein. It is intended, therefore, that the present invention not be defined by the specific embodiments described herein, but rather by the appended claims, which are intended to be construed in accordance with the following well-settled principles of claim construction: (a) Each claim should be given its broadest reasonable interpretation consistent with the specification; (b) Limitations should not be read from the specification or drawings into the claims (e.g., if the claim calls for "antenna", and the specification and drawings show a coil, the claim term "antenna" should not be limited to a coil, but rather should be construed to cover any type of antenna); (c) The words "comprising", "including", and "having" are always open-ended, irrespective of whether they appear as the primary transitional phrase of a claim or as a transitional phrase within an element or sub-element of the claim; (d) The indefinite articles "a" or "an" mean one or more; where, instead, a purely singular meaning is intended, a phrase such as "one", "only one", or "a single", will appear; (e) Words in a claim should be given their plain, ordinary, and generic meaning, unless it is readily apparent from the specification that an unusual meaning was intended; (f) an absence of the

US 6,775,372 B1

15

specific words "means for" connotes applicants' intent not to invoke 35 U.S.C. §112 (6) in construing the limitation; (g) Where the phrase "means for" precedes a data processing or manipulation "function," it is intended that the resulting means-plus-function element be construed to cover any, and all, computer implementation(s) of the recited "function"; (h) a claim that contains more than one computer-implemented means-plus-function element should not be construed to require that each means-plus-function element must be a structurally distinct entity (such as a particular piece of hardware or block of code); rather, such claim should be construed merely to require that the overall combination of hardware/firmware/software which implements the invention must, as a whole, implement at least the function(s) called for by the claim's means-plus-function element(s); (i) a means-plus-function element should be construed to require only the "function" specifically articulated in the claim, and not in a way that requires additional "functions" which may be described in the specification or performed in the preferred embodiment(s); (j) The existence of method claims that parallel a set of means-plus-function apparatus claims does not mean, or suggest, that the method claims should be construed under 35 U.S.C. §112 (6).

I claim:

1. A multi-stage data logging system comprising:

- a) a telecommunications ("telecom") stage receiving input from a plurality of communication channels;
- b) a recorder stage having one or more recorders, at least one recorder logging data associated with information transmitted on at least one of said plurality of communication channels;
- c) a distribution stage providing access to data logged in the recorder stage;
- d) a first interface linking the telecom and the recorder stages and a second interface linking the recorder and the distribution stages;

wherein at least two stages of the system are physically separable and in operation can be located wide distances apart.

2. The data logging system of claim 1 wherein the telecom stage comprises:

- a1) a first interface capturing signals from said plurality of communication channels;
- a2) one or more signal processors converting captured signals into formatted data; and
- a3) a second interface for transmitting said converted data to said recorder stage.

3. The data logging system of claim 2 wherein said one or more data processors provide data compression.

4. The data logging system of claim 2 wherein said one or more data processors encrypt the converted data.

5. The data logging system of claim 2 wherein the telecom stage further comprises at least one of: analog to digital signal converter and means for monitoring digital telephones.

6. The data logging system of claim 1 wherein the telecom stage provides time stamping of the received input.

7. The data logging system of claim 1 wherein the telecom stage provides authentication of signals from said plurality of input channels.

8. The data logging system of claim 1 wherein the recorder stage comprises a controller for directing and monitoring recorder stage operations, and each recorder comprises:

- b1) a first interface receiving data from the telecom stage;
- b2) a buffer for transitional data storage;
- b3) a random access storage device for data storage; and

16

b4) a second interface for transmitting stored data to the distribution stage.

9. The data logging system of claim 8 wherein the recorder stage still further comprises an archive storage device for archiving data.

10. The data logging system of claim 8, wherein the random access storage device is a hard disk.

11. The data logging system of claim 9 wherein said archive storage device is fixed.

12. The data logging system of claim 9 wherein said archive storage device is a RAID array.

13. The data logging system of claim 9 wherein said archive storage device is removable.

14. The data logging system of claim 1 wherein the distribution stage comprises:

- c1) a first interface receiving data from the recorder stage;
- c2) a controller for directing and monitoring distribution stage operations;
- c3) a buffer for transitional data storage; and
- c4) a second interface for distributing data to one or more output channels.

15. The data logging system of claim 1 wherein the distribution stage comprises an archive storage device for archiving data.

16. The data logging system of claim 15 wherein said archive storage device is fixed.

17. The data logging system of claim 15 wherein said archive storage device is a RAID array.

18. The data logging system of claim 15 wherein said archive storage device is removable.

19. The data logging system of claim 1 wherein the distribution stage comprises: an operating system software application and a computer capable of running said software application and accessing one or more remote serve computers.

20. The data logging system of claim 19 wherein said computer is connected to said one or more remote server computers via a local area network.

21. The data logging system of claim 19 wherein said computer is connected to said one or more remote server computers via an Internet protocol (I/P) network.

22. The data logging system of claim 1 wherein the recorder stage comprises at least one backup recorder, and the system further comprises means for detecting a malfunction in a recorder of the recorder stage, and means for automatically switching interface links from the detected malfunctioning recorder to said at least one backup recorder.

23. The data logging system of claim 1 wherein at least one of said first and second interfaces is network-based.

24. The data logging system of claim 1 wherein at least one of said first and second interfaces is a four-wire interface.

25. The data logging system of claim 1, wherein the telecom stage comprises at least one of: voltage sensing, current sensing or external signals to trigger recording of data from a communication channel.

26. The data logging system of claim 1, wherein input from the plurality of communication channels comprises one or more of: voice, image or facsimile data types, and said one or more data processors are configured to process said data types.

27. The data logging system of claim 26, wherein output of said one or more data processors is compatible with the G 723.1 or G 722 international voice compression standards.

28. The data logging system of claim 26, wherein said one or more data processors are re-programmable.

29. The data logging system of claim 1, wherein input from different communication channels is stored as a plurality of files having unique record addresses.

30. The data logging system of claim 29, wherein each file has associated call record information.

US 6,775,372 B1

17

31. The data logging system of claim 29, wherein the recorder stage is implemented as a server supporting a plurality of file access protocols.

32. The data logging system of claim 1, wherein the distribution stage is implemented as a network server.

33. The data logging system of claim 32, wherein the network server is a Web server.

34. The data logging system of claim 32, wherein the network server is a file server.

35. The data logging system of claim 33, wherein users can access the Web server through a browser.

36. The system of claim 1, wherein at least two stages are located in different rooms of the same building.

37. The system of claim 1, wherein the recorder stage and the distribution stage are located in different cities.

38. A multi-stage data logging system comprising:

a) a first means for receiving signals from one or more communication channels;

b) a second means for logging data associated with received signals;

c) a third means for retrieving logged data and distributing retrieved data to one or more output channels;

wherein at least two of said first, second, and third means are physically separable and can operate wide distances apart.

39. The data logging system of claim 38 further comprising an archive storage device for archiving data from said one or more communication channels.

40. The data logging system of claim 38 wherein received signals from said one or more communication channels are voice signals, and the second means further comprises means for recording call information about the received voice signals.

41. The data logging system of claims 1 or 38 wherein linking of at least one of the telecom and recorder stages, and recorder and distribution stages is provided over a communications network.

42. The data logging system of claim 41, wherein the communications network is one or more of: private communication lines, public switched telephone network, corporate intranet, the Internet or a combination thereof.

43. A data logger, comprising:

a telecommunication device receiving input from a plurality of communication channels;

a processor converting the received input to one or more data formats;

a memory for logging information about the received input, the information comprising data converted to at least one data format;

18

a communication path to a communications network; and a server having access to the memory via the communications network for transferring logged data from one or more of said plurality of communication channels via the communications network to at least one remote user.

44. The data logger of claim 43 wherein the server is a Web server and the communications network is the Internet.

45. A method for operating a multi-stage data logging system having: a telecom stage receiving input from a plurality of channels; a recorder stage having two or more recorders, at least one recorder storing data associated with input received from the plurality of channels and at least one backup recorder; a distribution stage providing access to data stored in the recorder stage; and a first interface linking the telecom and said one or more recorders of the recorder stages and a second interface linking the recorder and the distribution stages; the method comprising:

detecting a malfunctioning recorder in the recorder stage;

automatically switching interface links from the detected malfunctioning recorder to said backup recorder to ensure uninterrupted operation of the system; and

without disrupting the operation of the system replacing the detected malfunctioning recorder with a functioning recorder.

46. A method for increasing the recording capacity of an operating multi-stage data logging system having: a telecom stage having telecom blocks capturing input from at most N input channels; a recorder stage having one or more recorders, said recorders having maximum recording capacity of M ($M \leq N$) channels; a distribution stage providing access to data stored in the recorder stage; a first network-based or four-wire-based interface linking the telecom and the recorder stages; and a second interface linking the recorder and the distribution stages; the method comprising:

(a) without disrupting the operation of the system attaching to said first interface at least one recorder so that the combined capacity of the recorders in the recorder stage is equal to or exceeds N channels

(b) without disrupting the operation of the system, attaching to said first interface at least one additional telecom block so that the system can capture $P > N$ input channels; and

(c) repeating step (a) until the combined capacity of the recorders in the recorder stage is equal to or exceeds P channels.

* * * * *

EXHIBIT G

U.S. Patent

Aug. 31, 2004

Sheet 1 of 29

US 6,785,370 B2

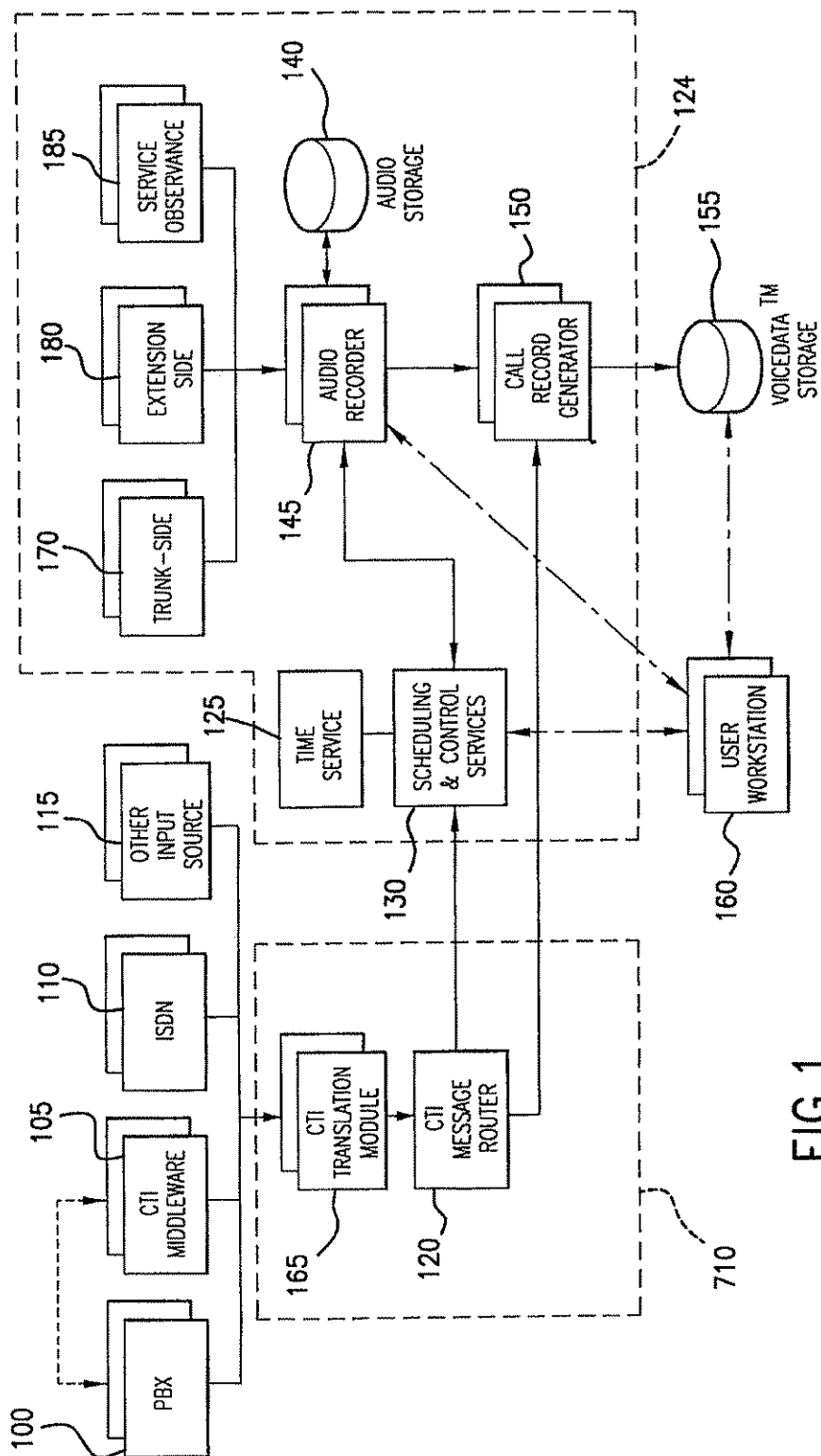


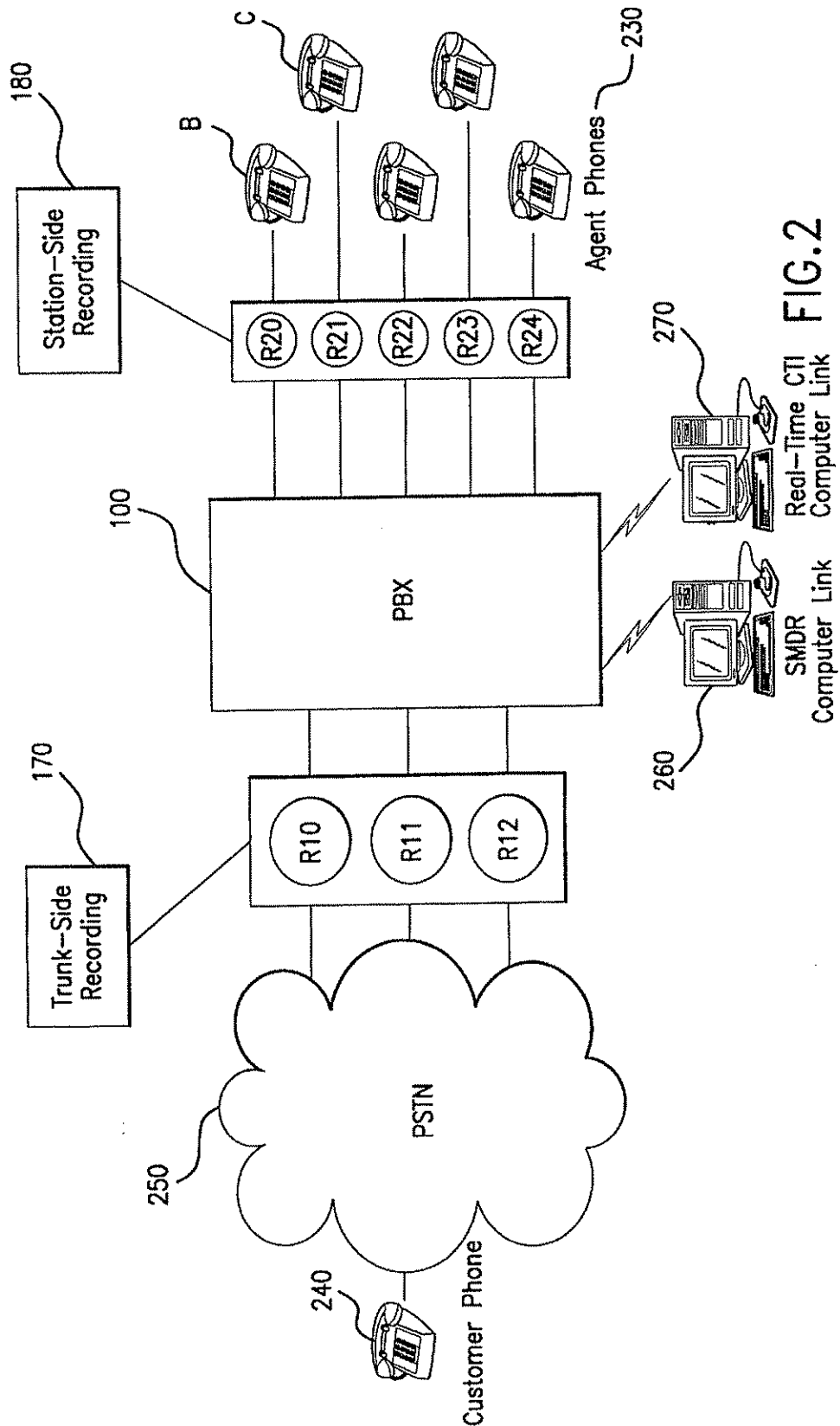
FIG.1

U.S. Patent

Aug. 31, 2004

Sheet 2 of 29

US 6,785,370 B2



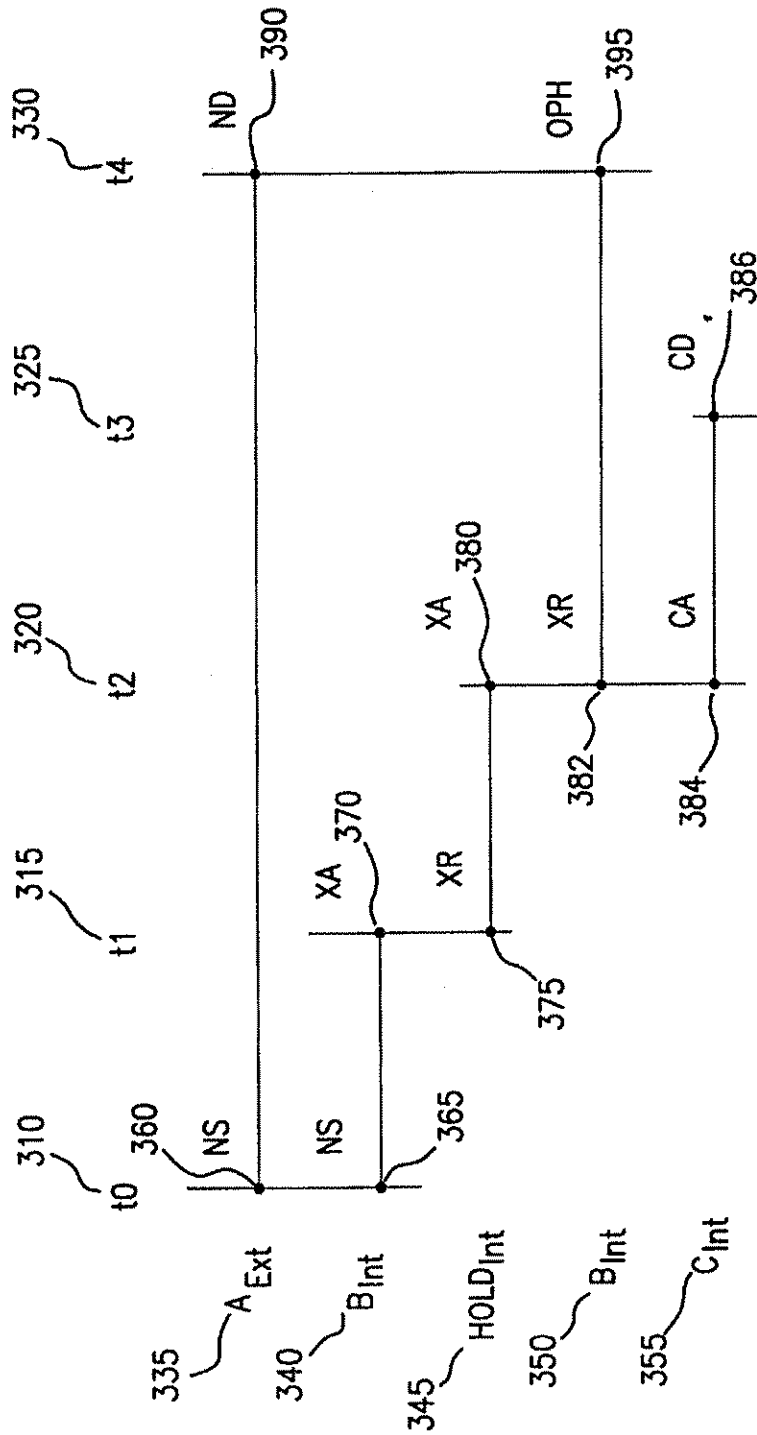


FIG. 3

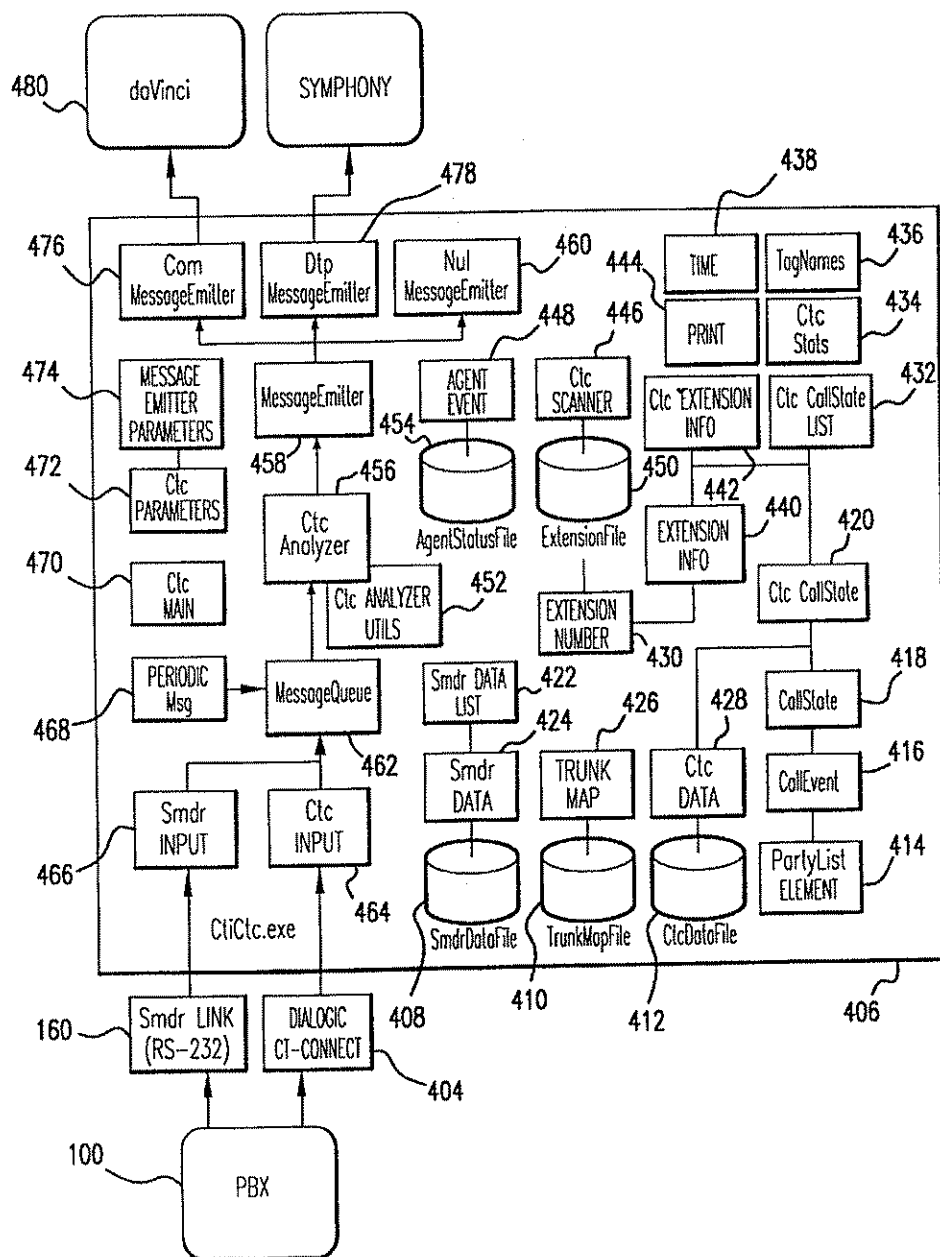


FIG.4

U.S. Patent

Aug. 31, 2004

Sheet 5 of 29

US 6,785,370 B2

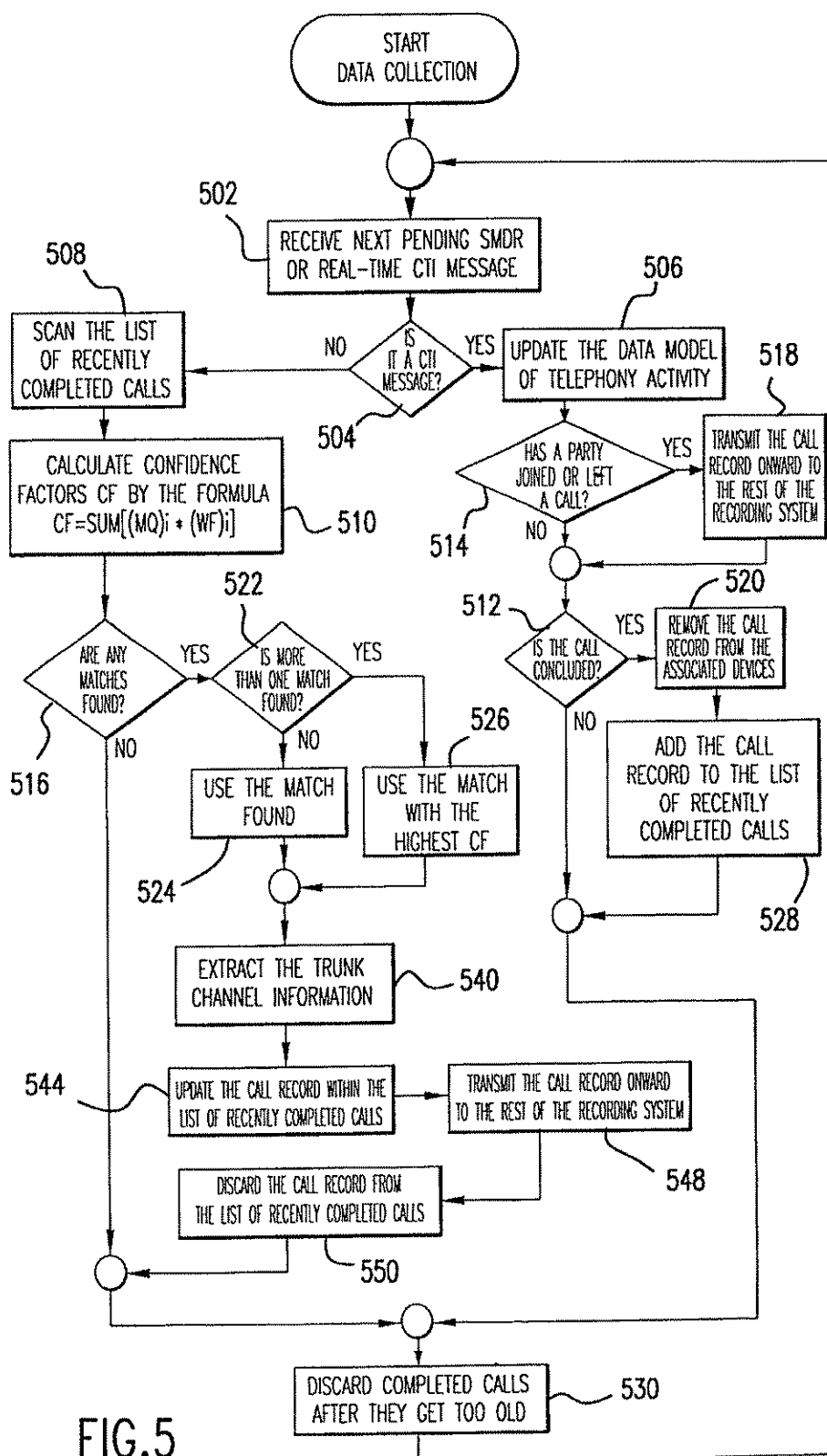


FIG. 5

U.S. Patent

Aug. 31, 2004

Sheet 6 of 29

US 6,785,370 B2

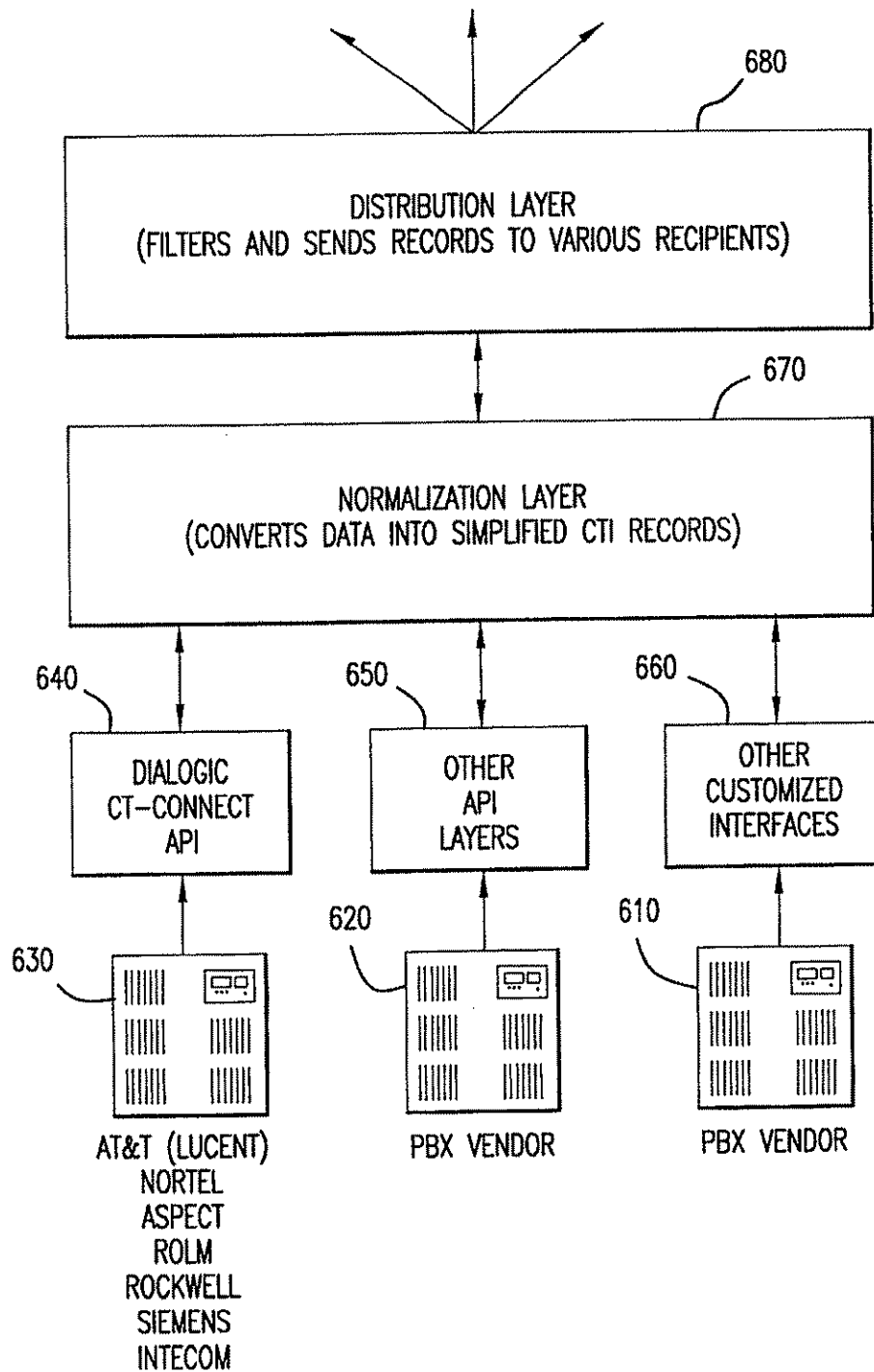


FIG.6

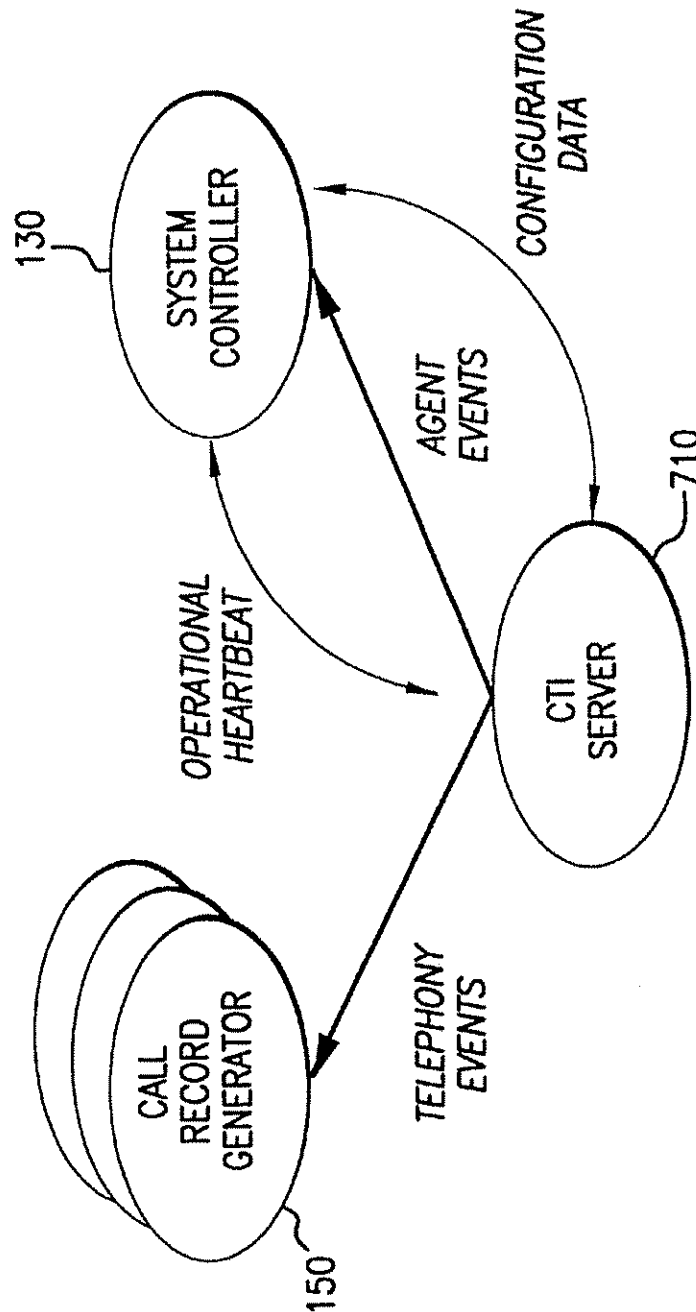


FIG. 7

U.S. Patent

Aug. 31, 2004

Sheet 8 of 29

US 6,785,370 B2

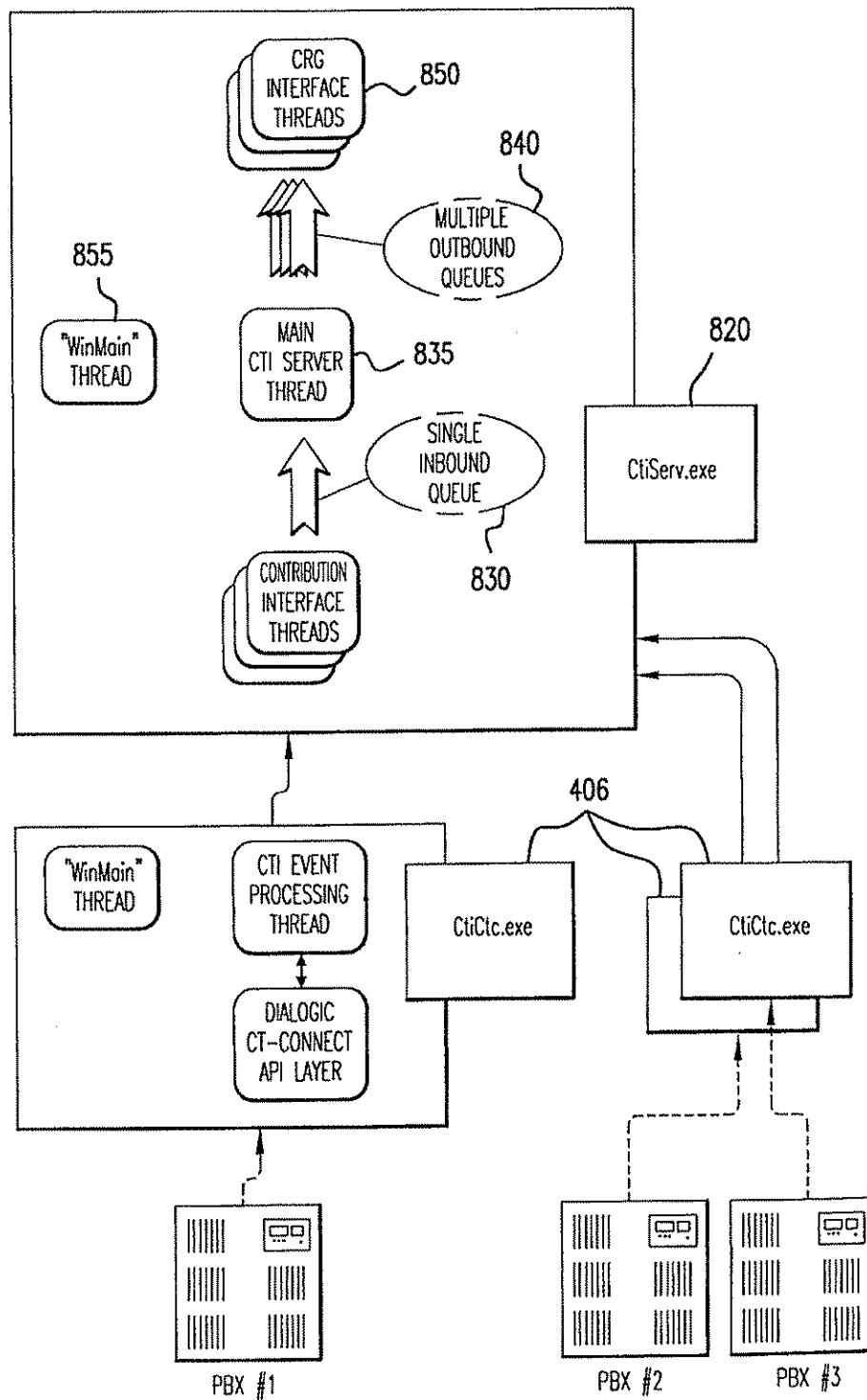


FIG. 8

U.S. Patent

Aug. 31, 2004

Sheet 9 of 29

US 6,785,370 B2

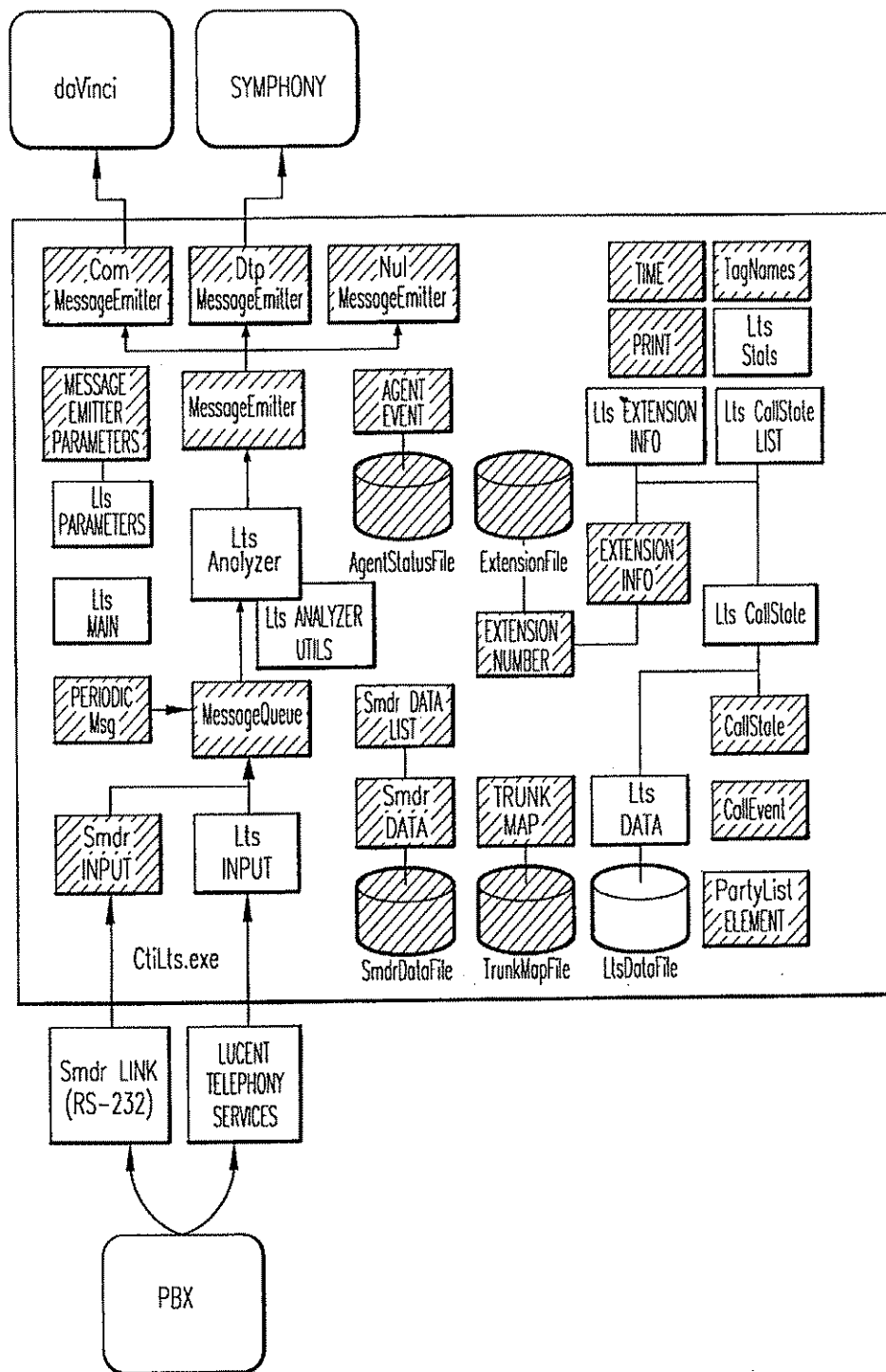
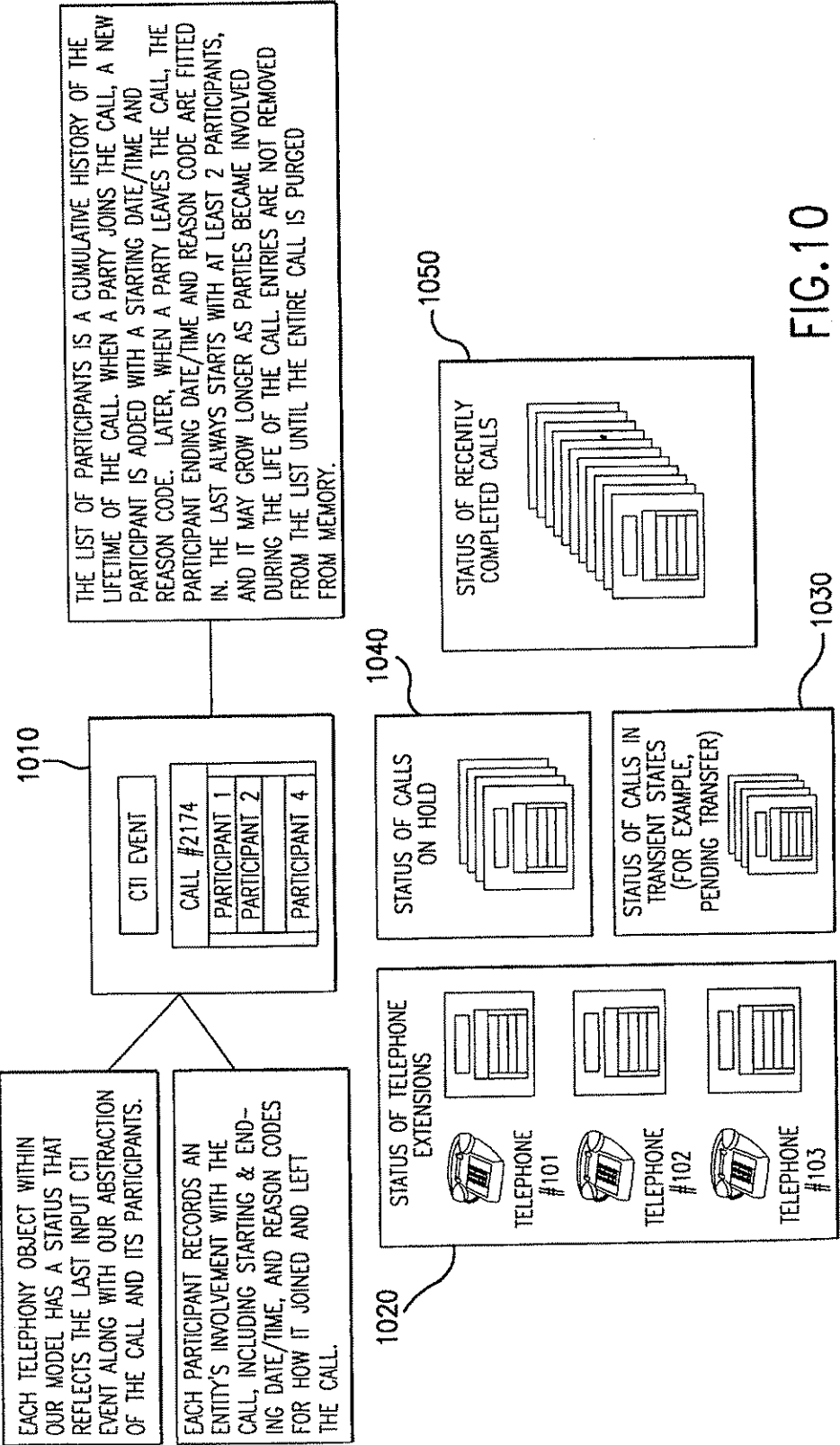


FIG.9



U.S. Patent

Aug. 31, 2004

Sheet 11 of 29

US 6,785,370 B2

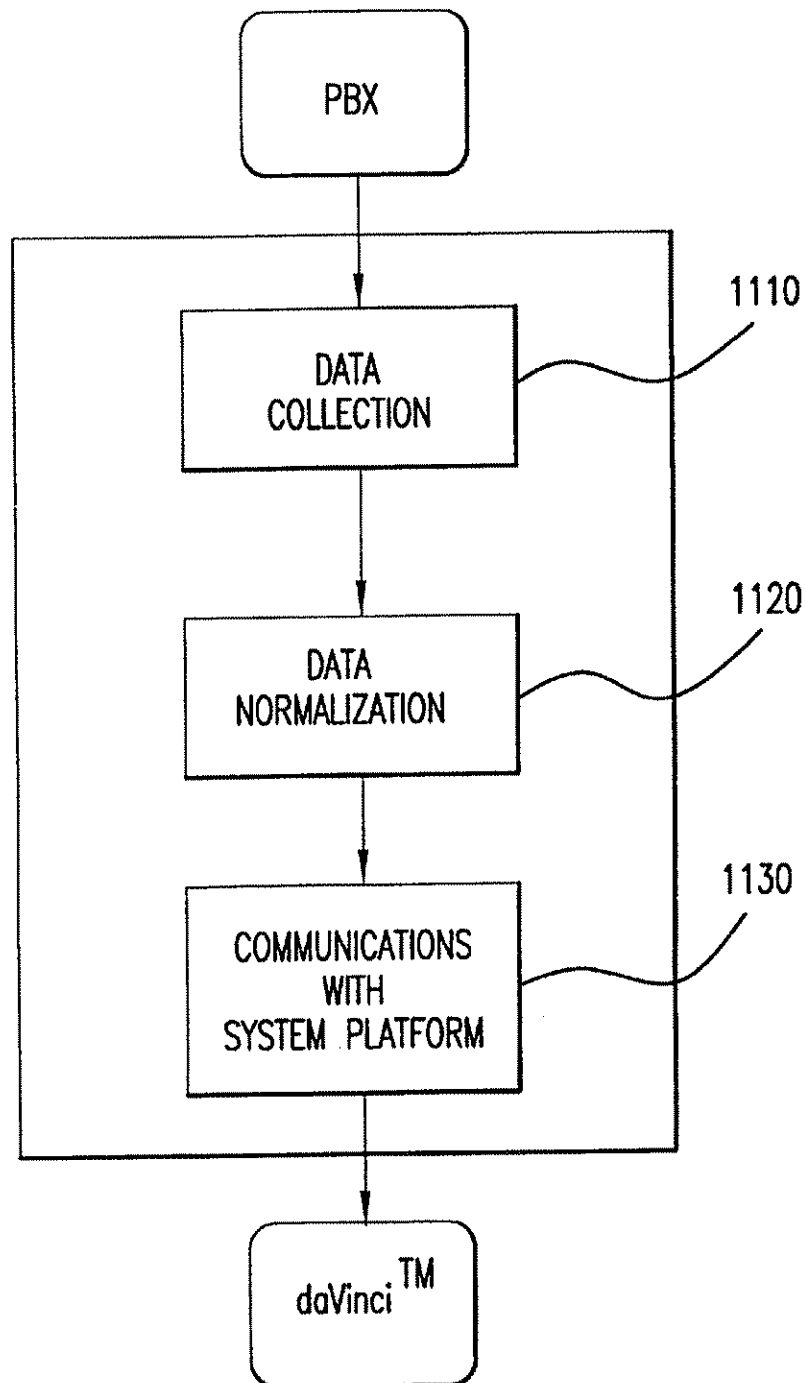


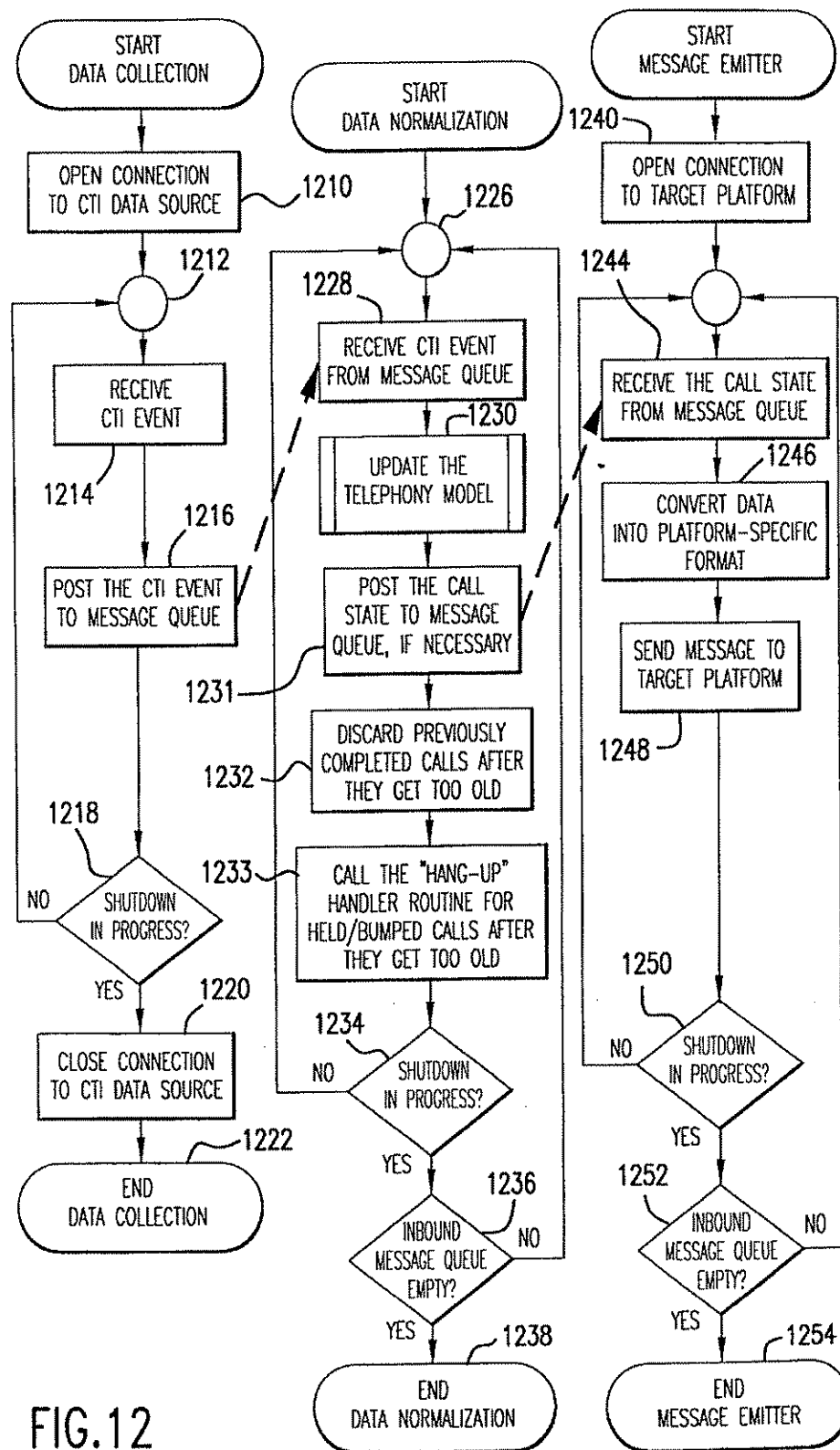
FIG. 11

U.S. Patent

Aug. 31, 2004

Sheet 12 of 29

US 6,785,370 B2



U.S. Patent

Aug. 31, 2004

Sheet 13 of 29

US 6,785,370 B2

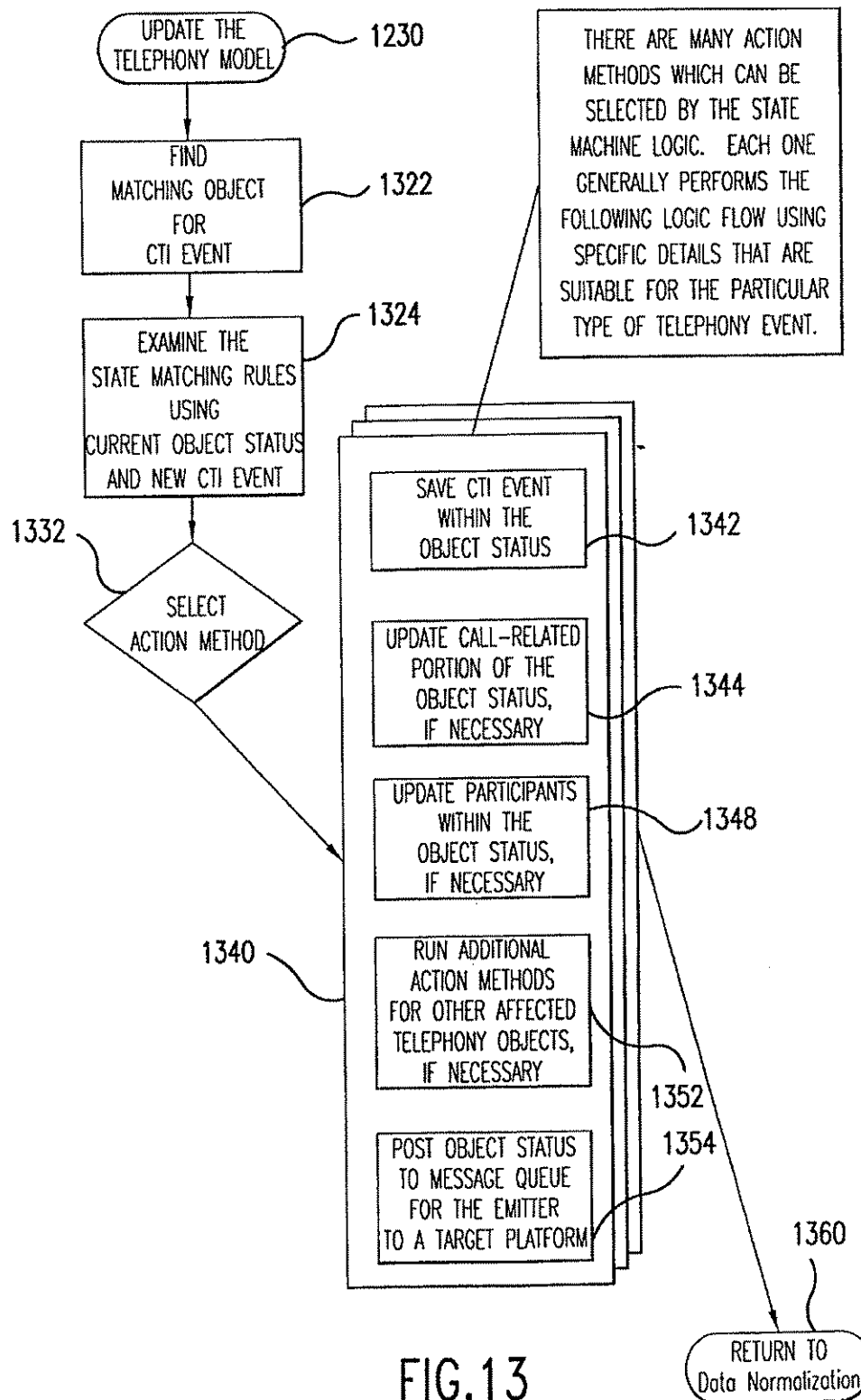
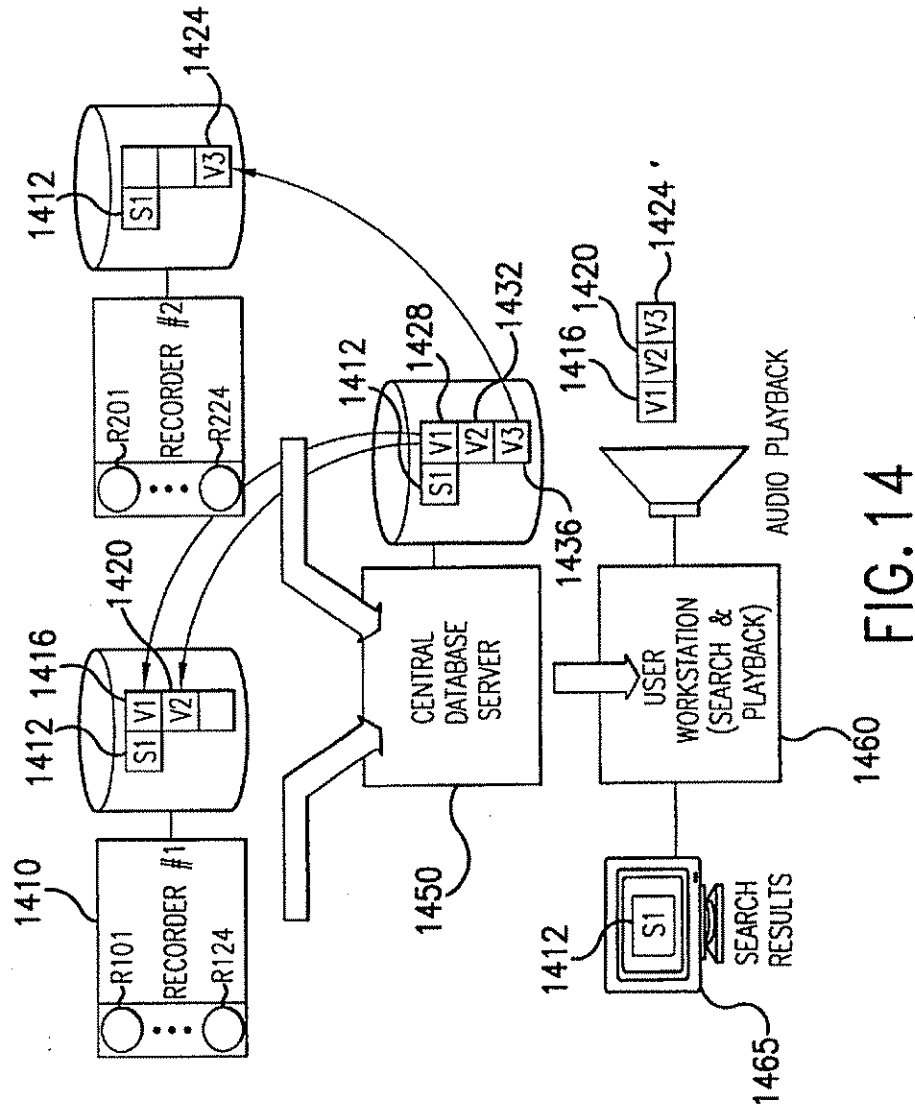


FIG. 13



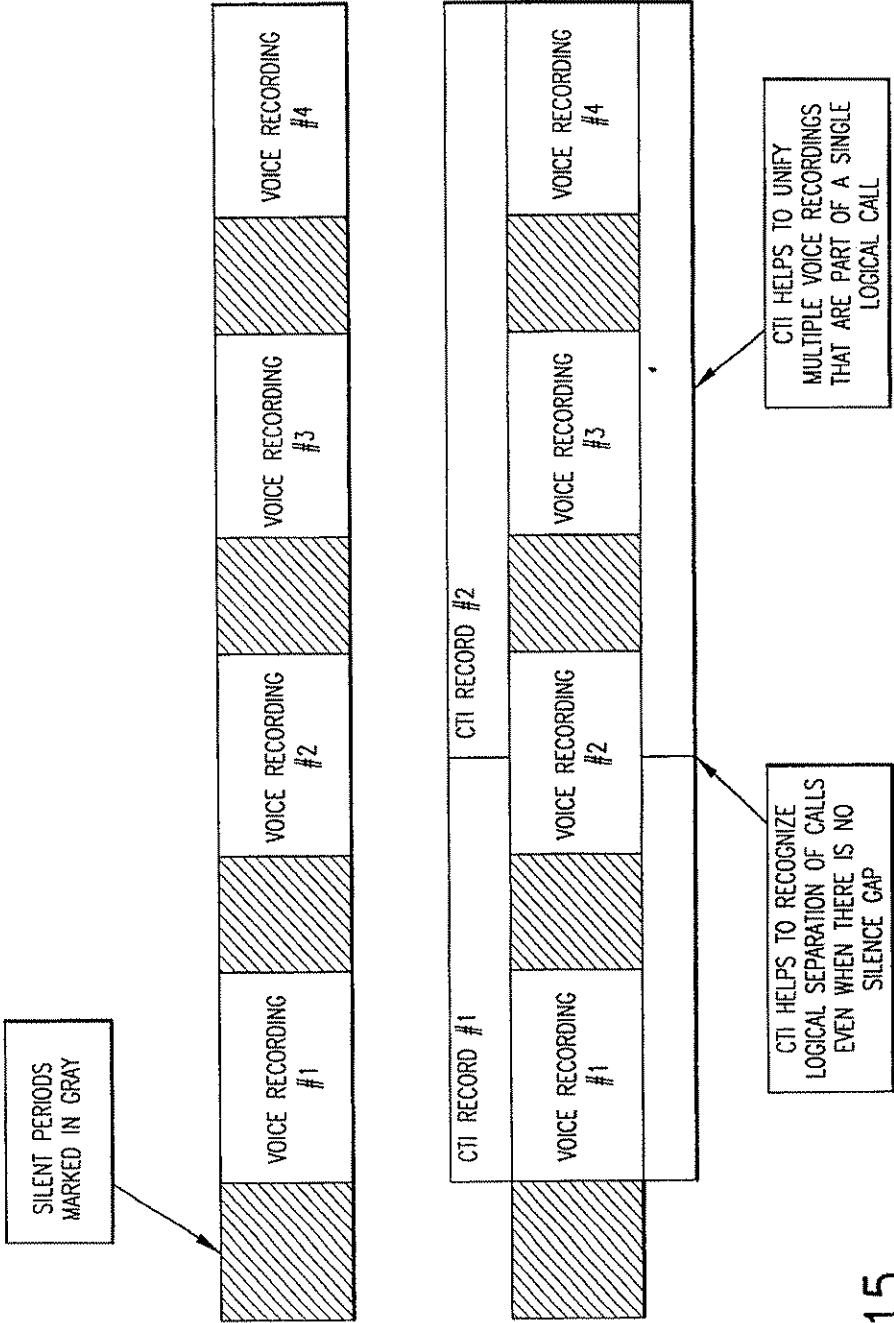


FIG.15

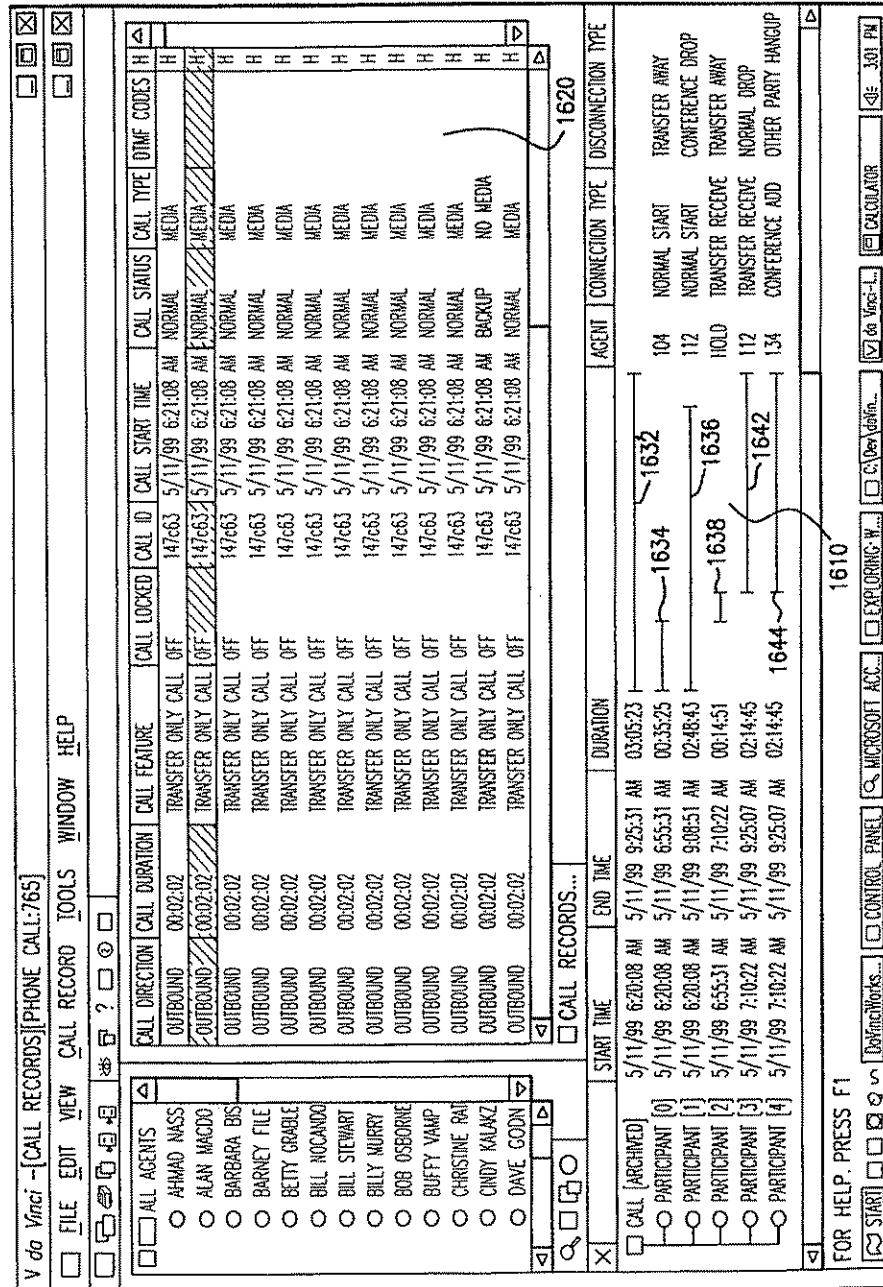


FIG. 16

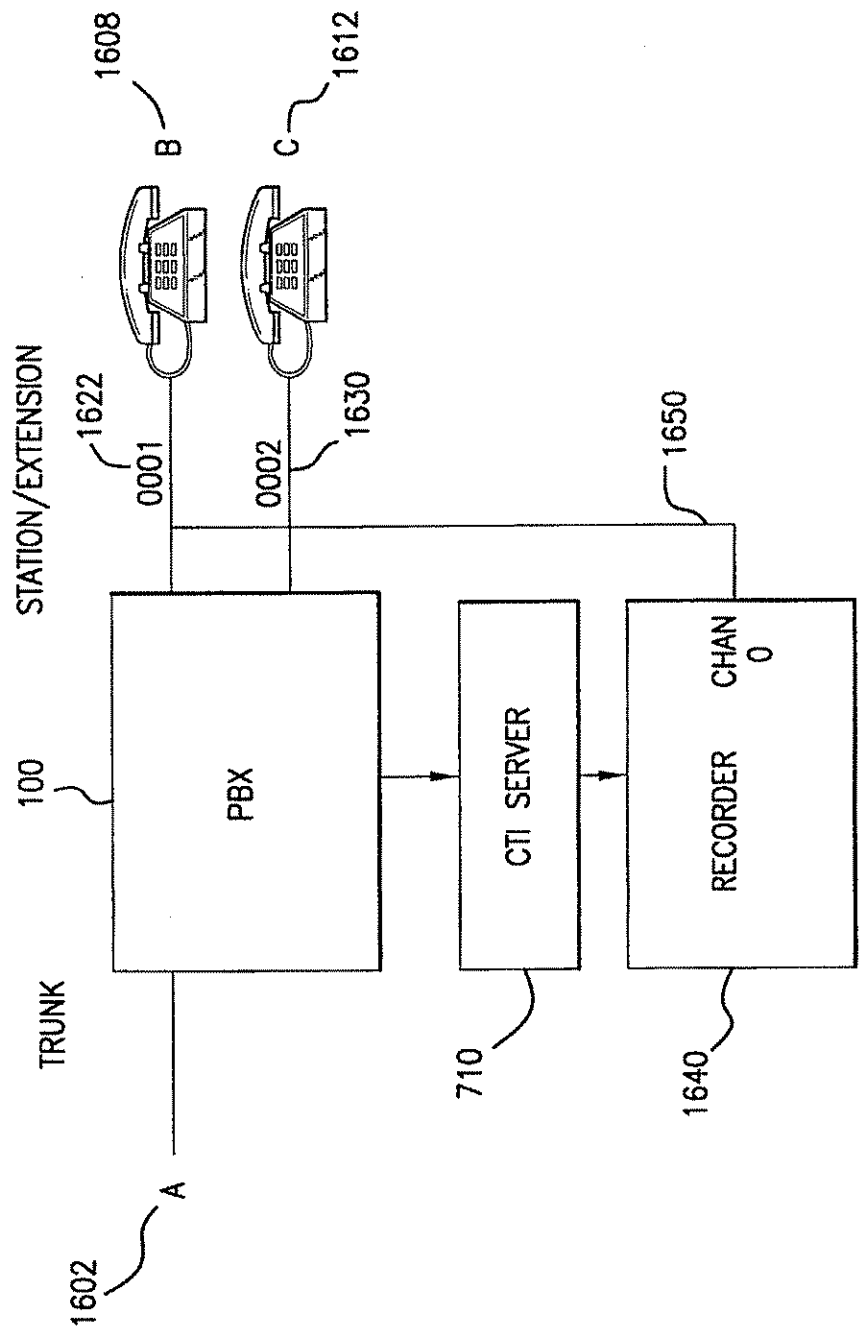


FIG.16A

U.S. Patent

Aug. 31, 2004

Sheet 18 of 29

US 6,785,370 B2

AgentId	EXTENSION	LOCATION	START TIME	END TIME	CONNECT REASON	DISCONNECT REASON
A		EXTERNAL	t_1	t_7	NORM START	NORM DROP
B	0001	INTERNAL	t_1	t_4	NORM START	Xfr AWAY
HOLD		INTERNAL	t_4	t_6	Xfr Rec	Xfr AWAY
C	0002	INTERNAL	t_6	t_7	Xfr Rec	OTHER PARTY HANGUP

FIG. 16B

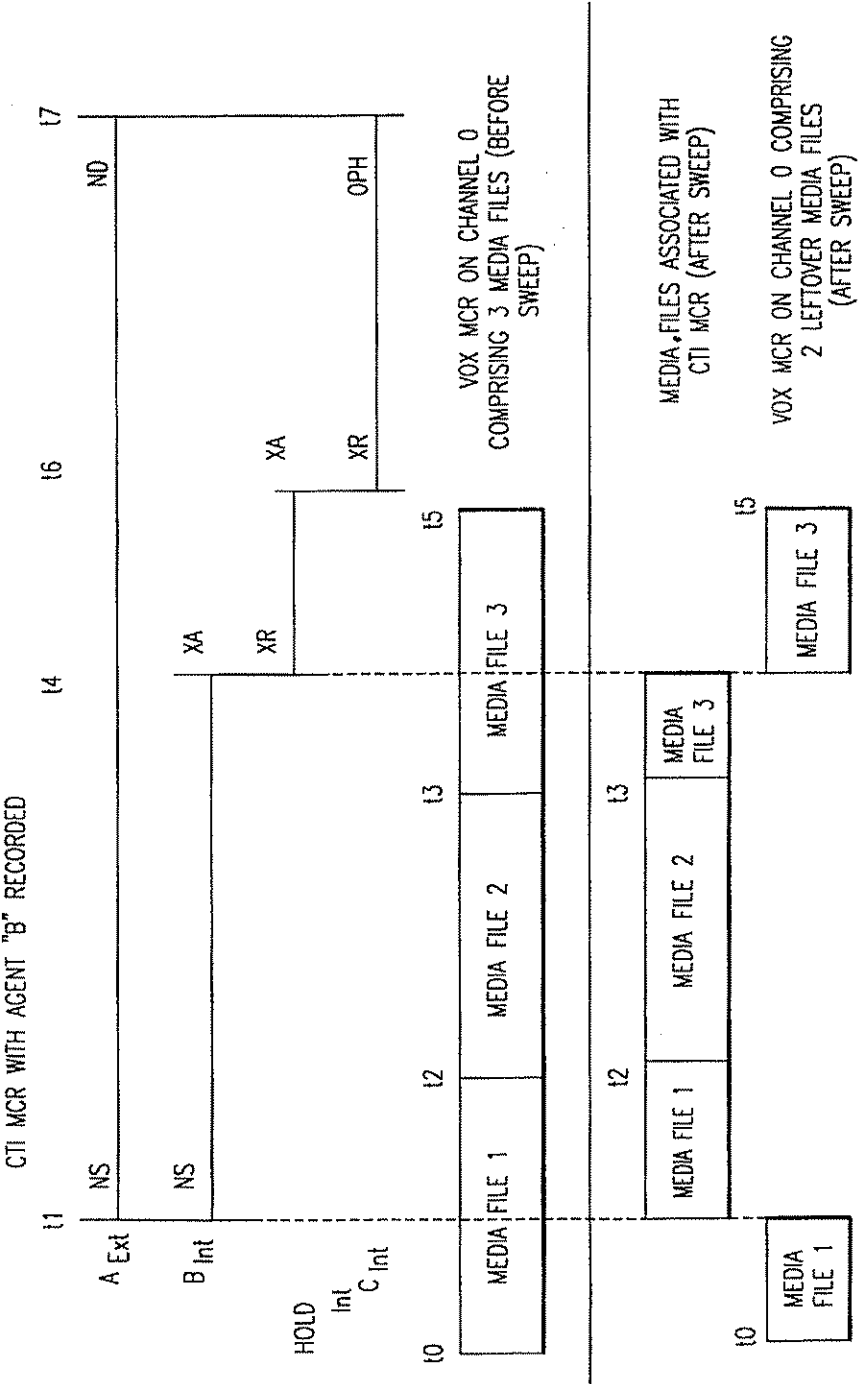


FIG.17

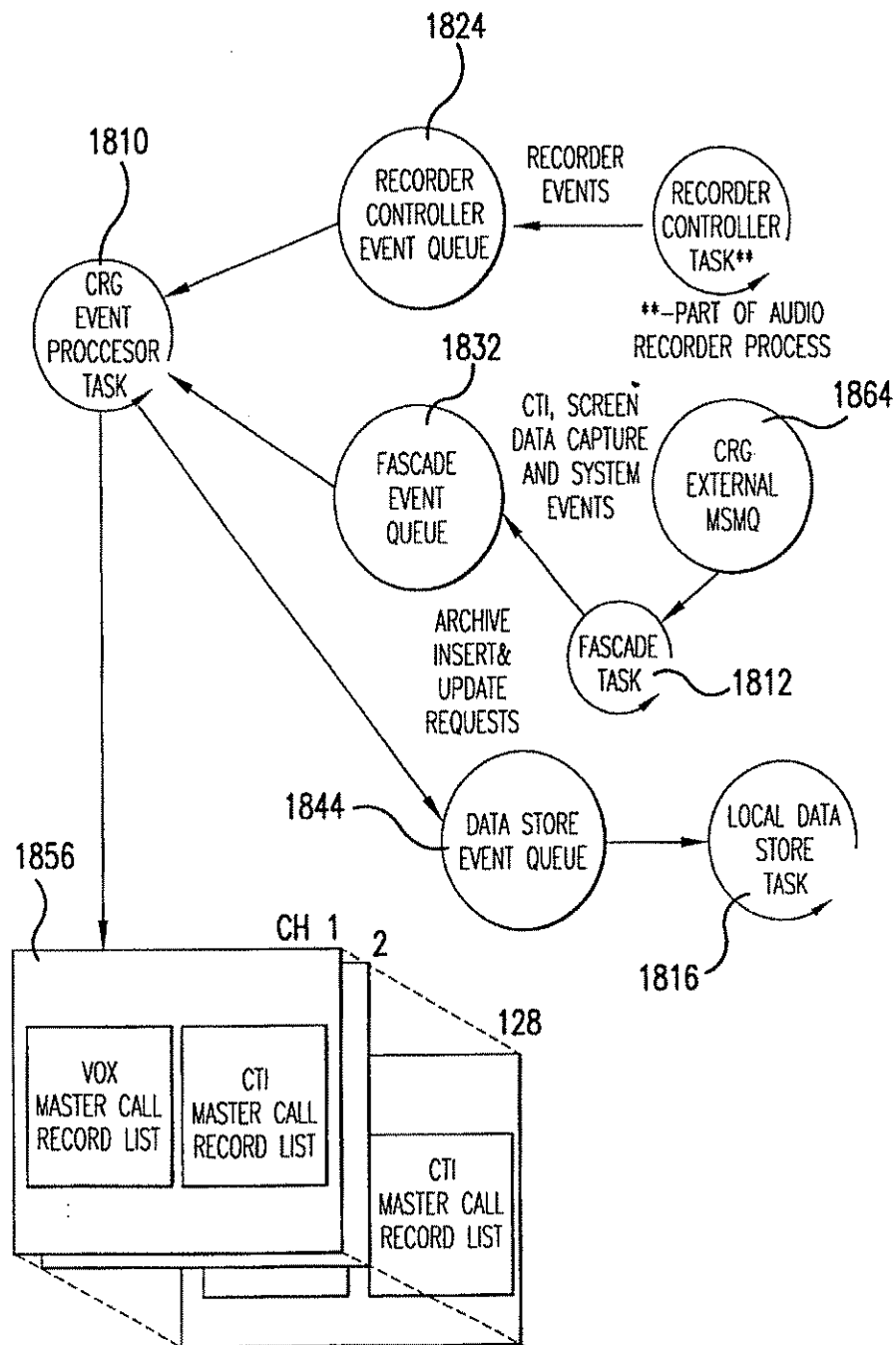


FIG. 18

U.S. Patent

Aug. 31, 2004

Sheet 21 of 29

US 6,785,370 B2

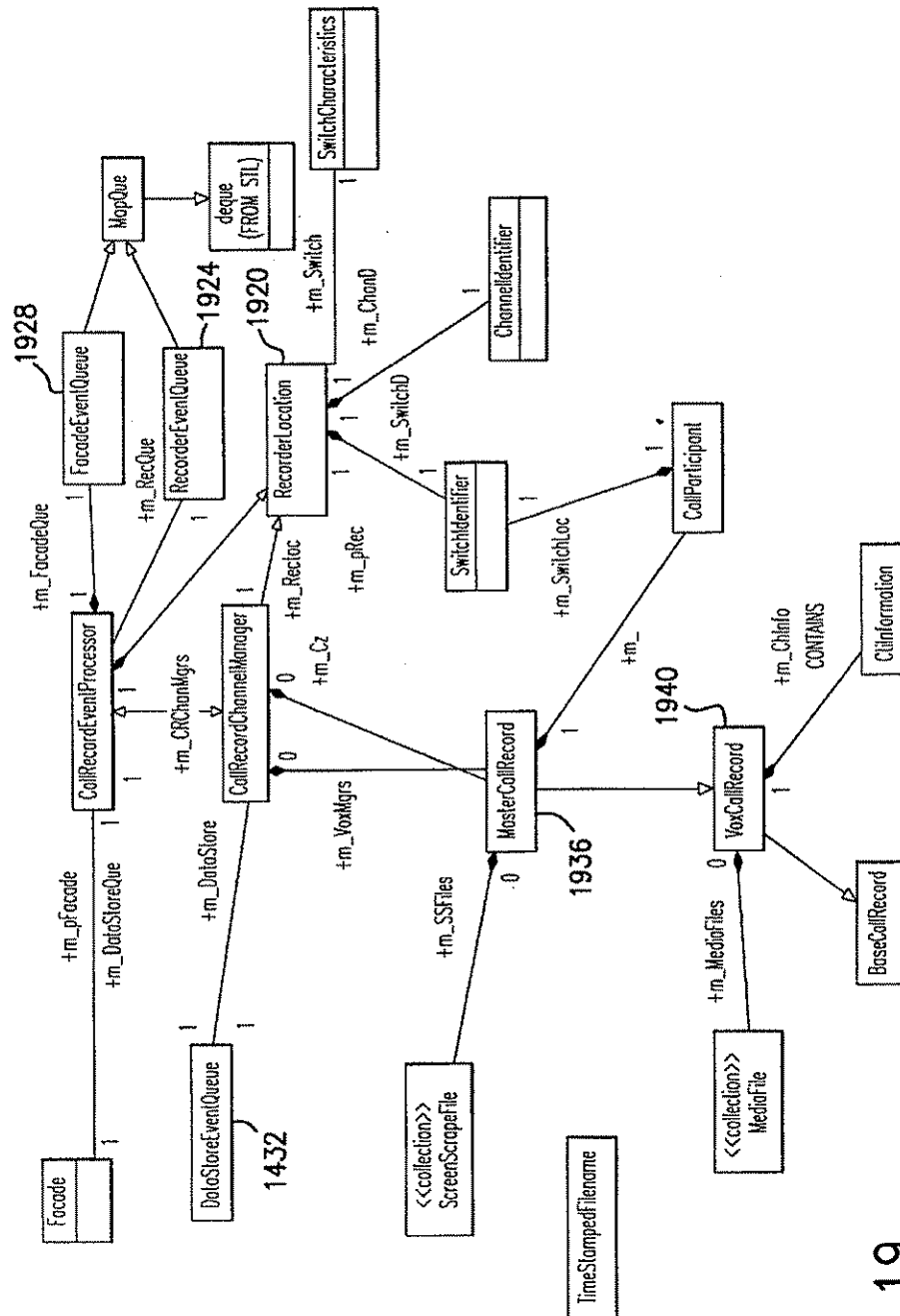


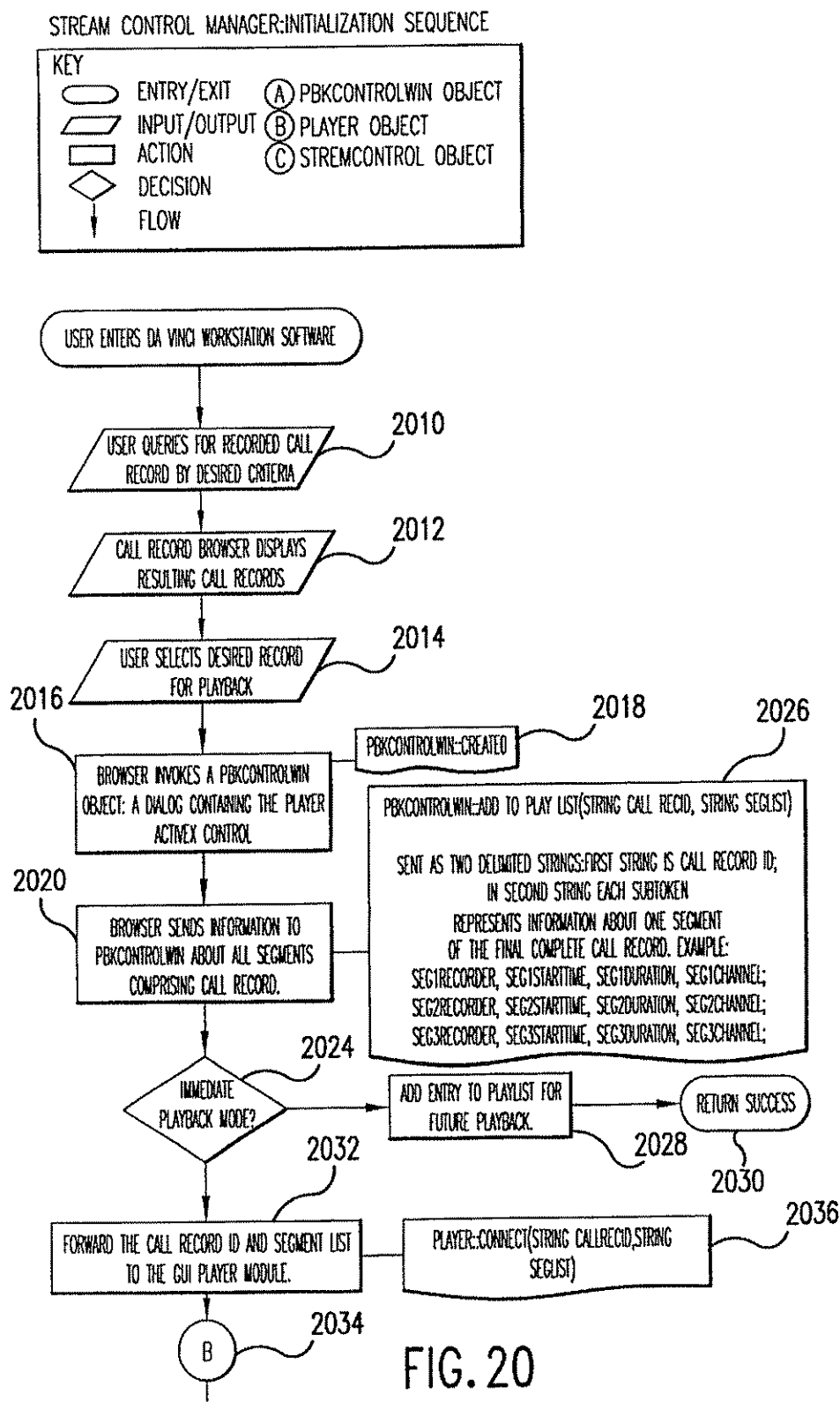
FIG. 19

U.S. Patent

Aug. 31, 2004

Sheet 22 of 29

US 6,785,370 B2



U.S. Patent

Aug. 31, 2004

Sheet 23 of 29

US 6,785,370 B2

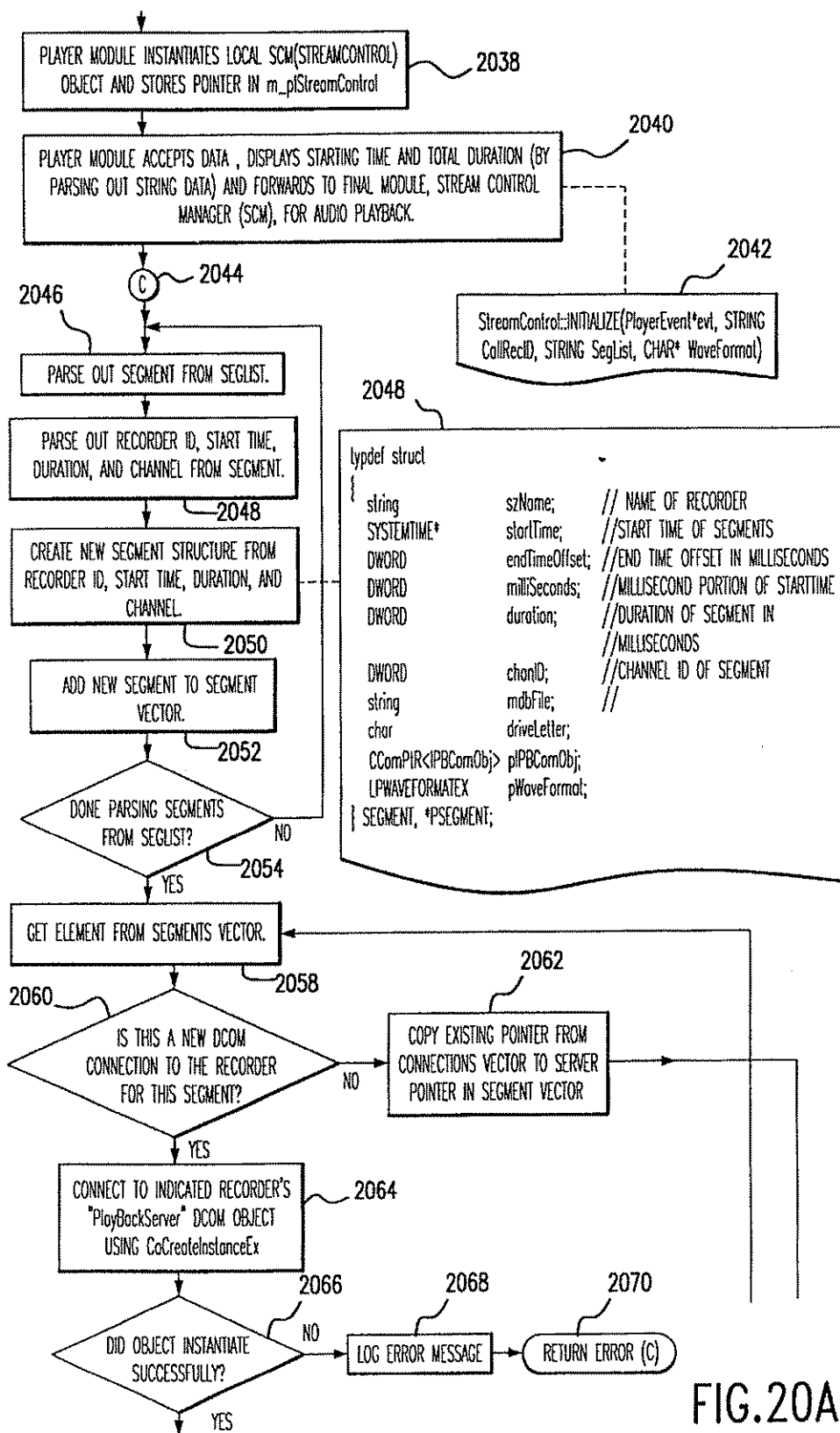


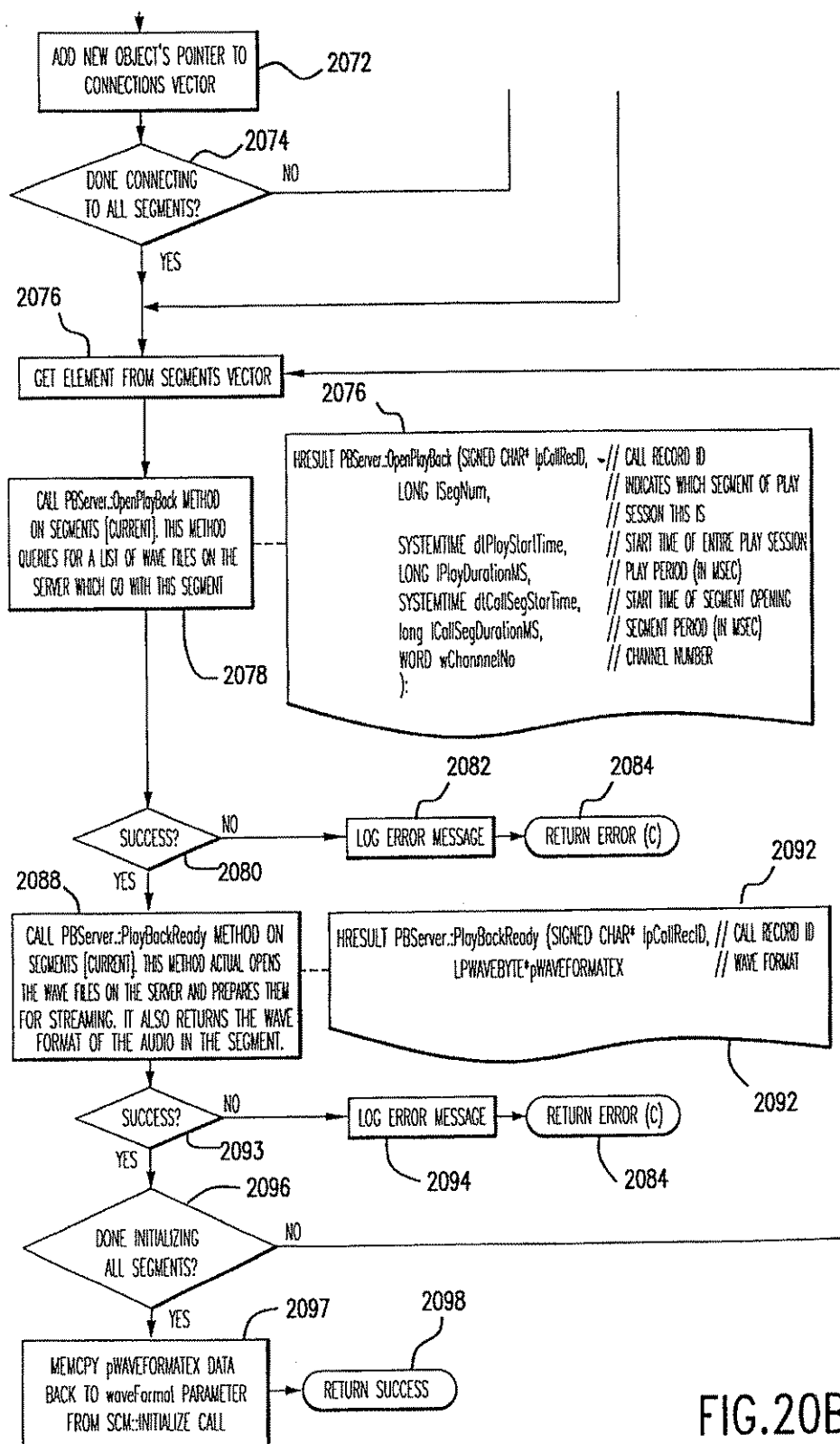
FIG. 20A

U.S. Patent

Aug. 31, 2004

Sheet 24 of 29

US 6,785,370 B2



U.S. Patent

Aug. 31, 2004

Sheet 25 of 29

US 6,785,370 B2

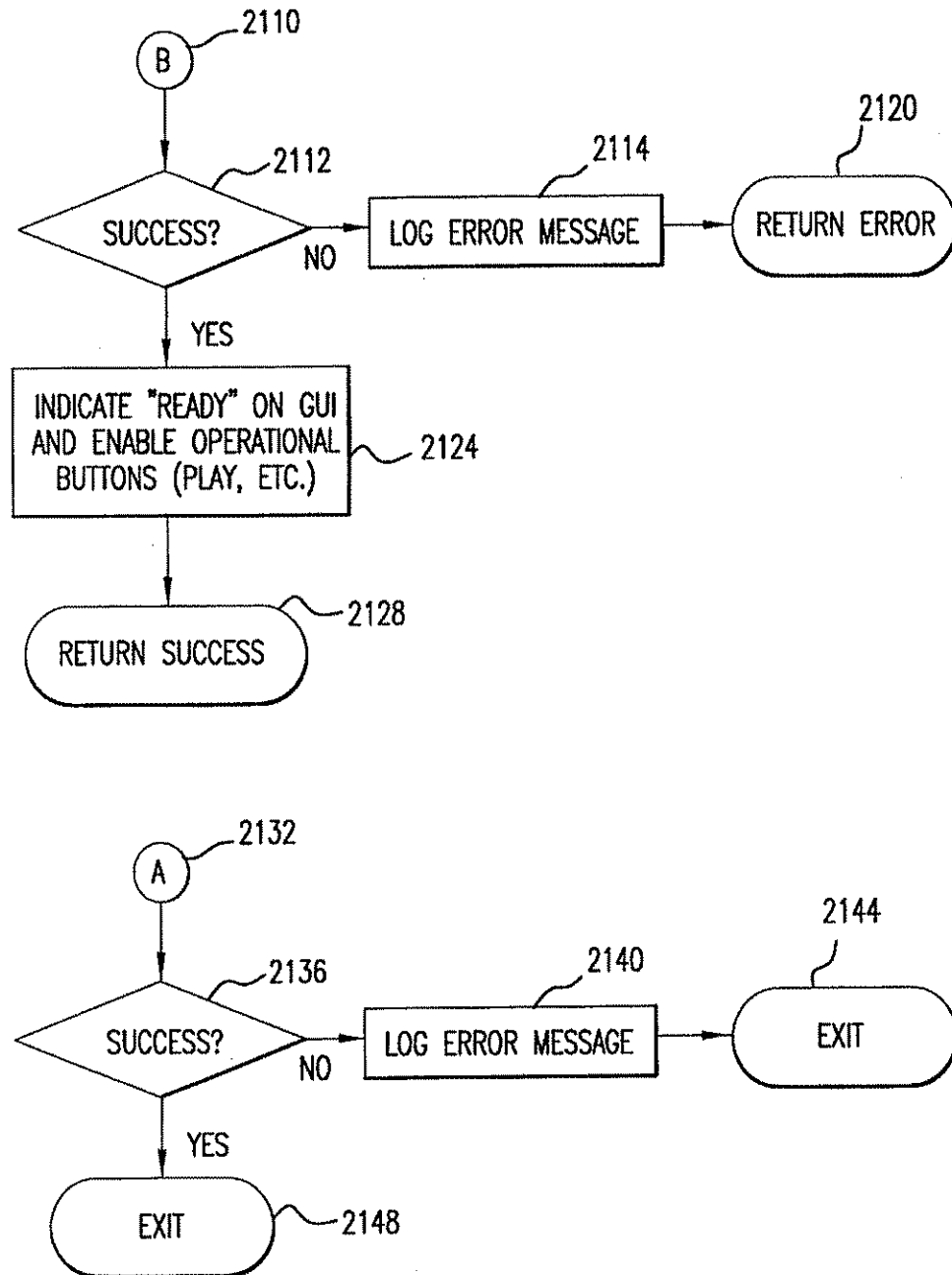


FIG.21

U.S. Patent

Aug. 31, 2004

Sheet 26 of 29

US 6,785,370 B2

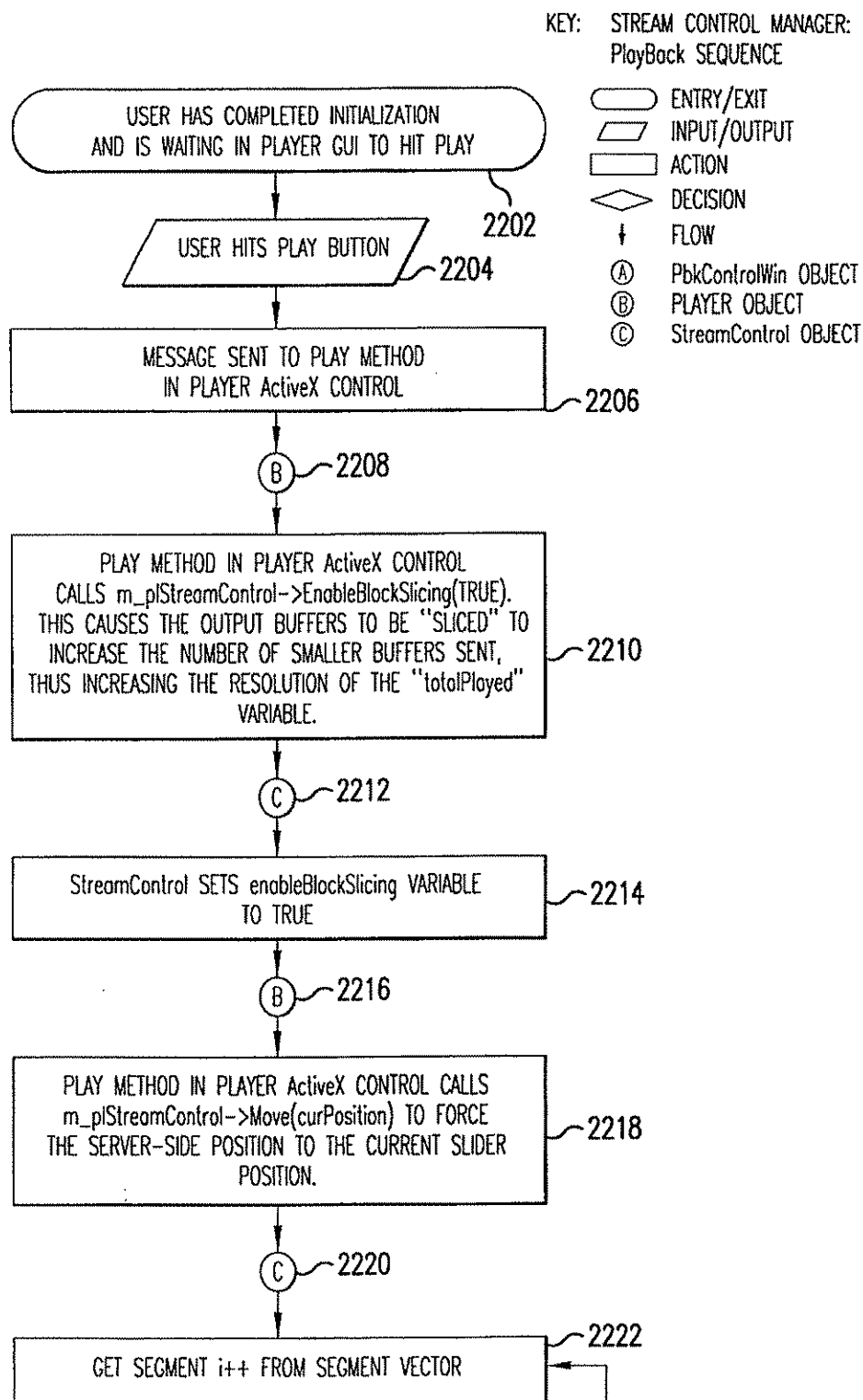


FIG.22

U.S. Patent

Aug. 31, 2004

Sheet 27 of 29

US 6,785,370 B2

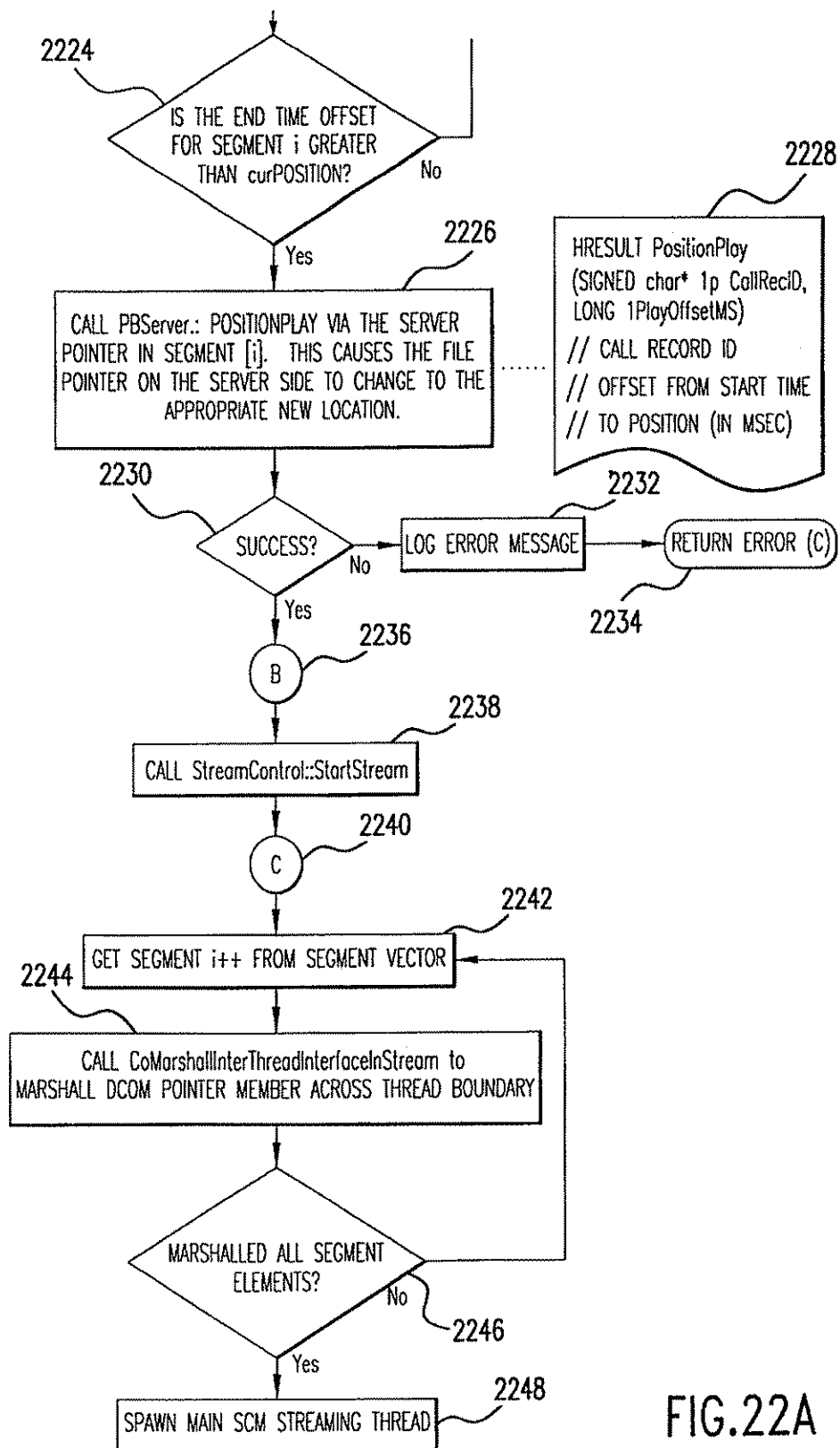


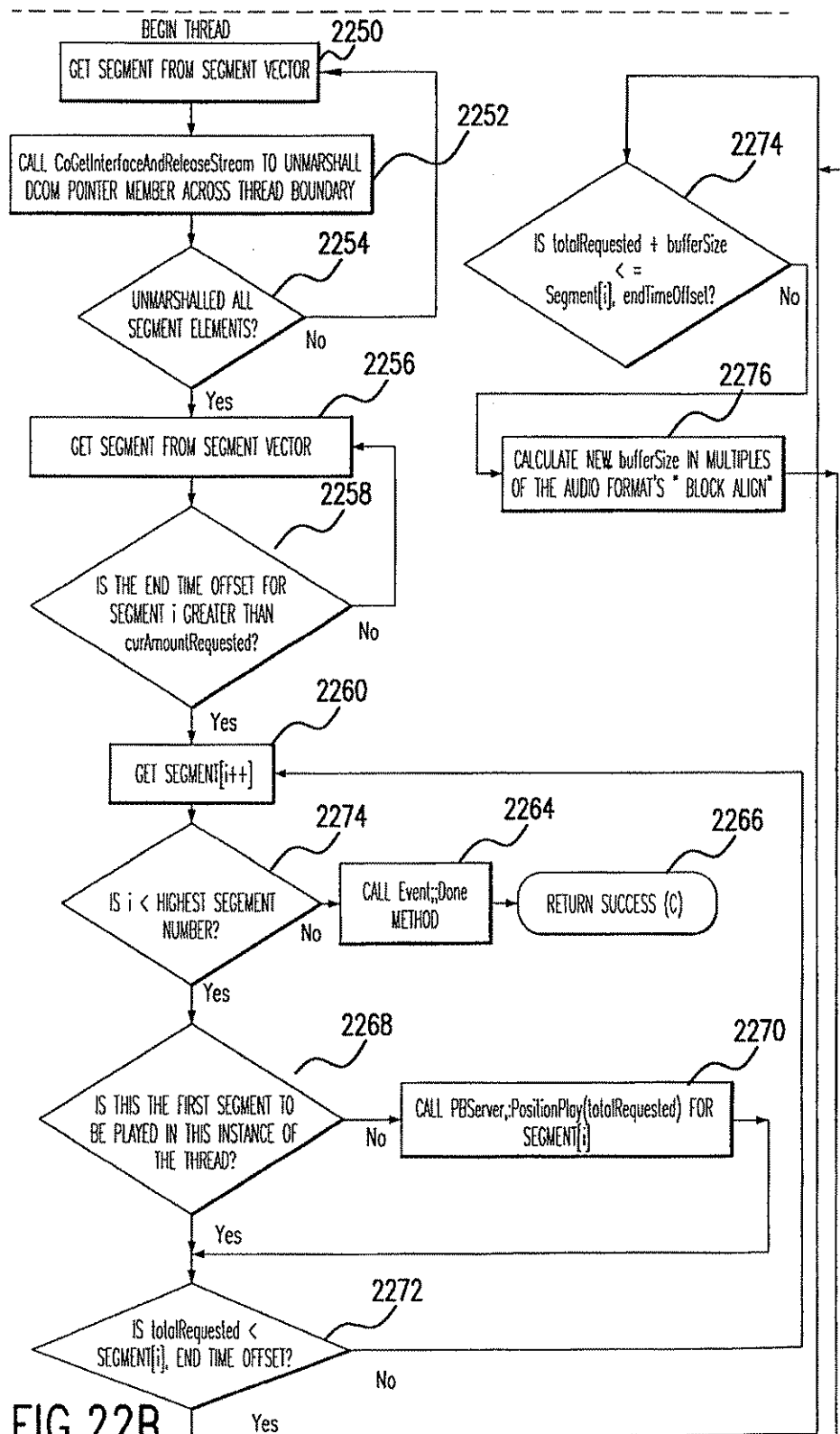
FIG. 22A

U.S. Patent

Aug. 31, 2004

Sheet 28 of 29

US 6,785,370 B2



U.S. Patent

Aug. 31, 2004

Sheet 29 of 29

US 6,785,370 B2

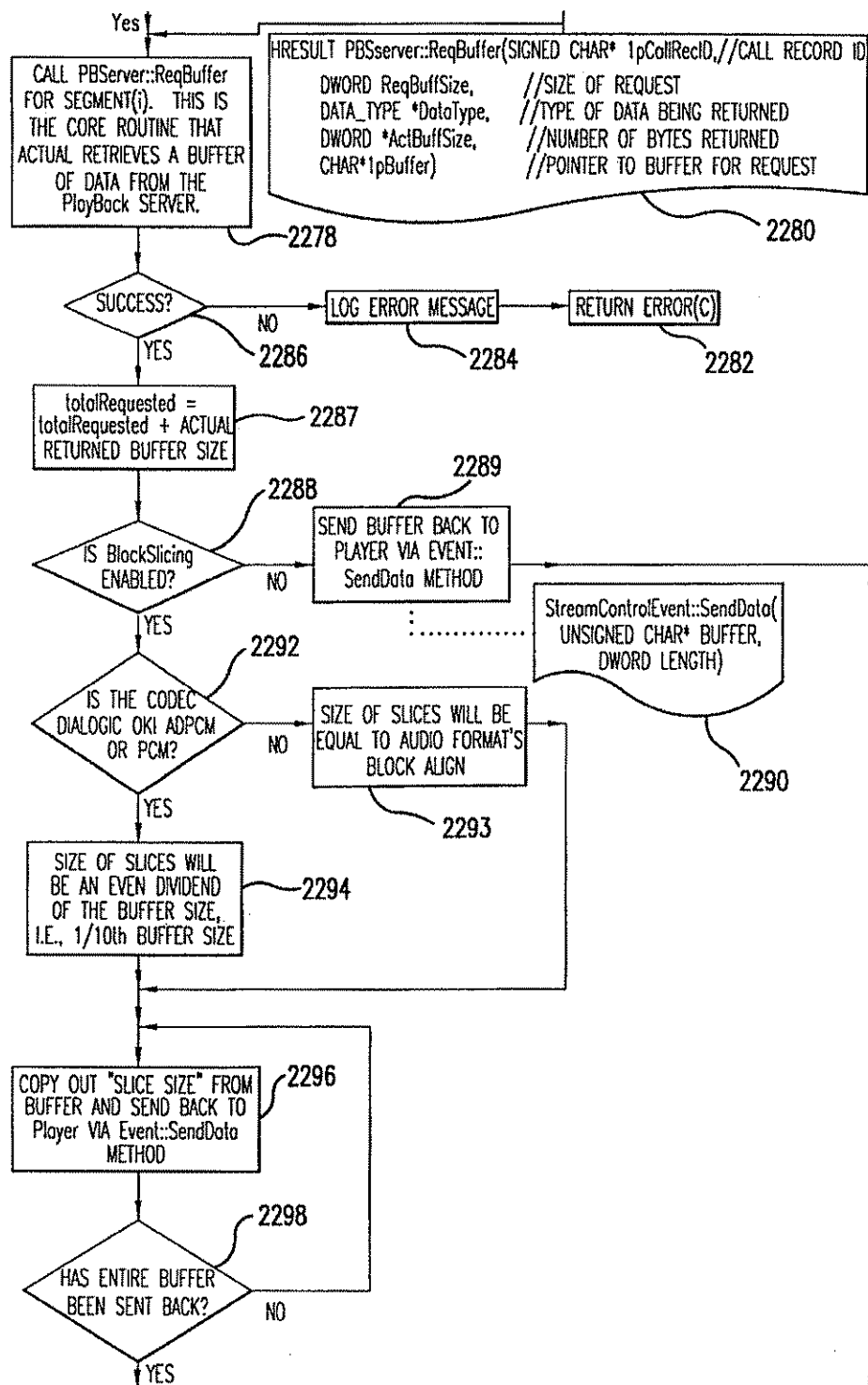


FIG. 22C

US 6,785,370 B2

1

SYSTEM AND METHOD FOR INTEGRATING CALL RECORD INFORMATION

This is a continuation of Application Ser. No. 09/328,294
filed Jun. 8, 1999 now U.S. Pat. No. 6,252,946.

FIELD OF THE INVENTION

This invention relates generally to computer-aided data
recording. In particular it relates to computer-aided moni-
toring and recording of telephone calls.

BACKGROUND OF THE INVENTION

Telephone call monitoring systems are used in a variety of
contexts, including emergency dispatch centers and com-
mercial call centers. In many currently available call moni-
toring systems, a multitude of audio input sources
("channels") are monitored and recorded by a single hard-
ware unit, and the audio recordings are saved and organized
according to the input channel, date, time and duration. The
capacity of the recording unit can be expanded to handle a
larger number of channels by combining several recording
units into a system using a local area network (LAN).
Because retrieval is only possible using basic search criteria
(recording unit, channel, date, time and duration), it is often
difficult to locate a particular audio recording that is of
interest. When there is a need to search for a recording
according to search criteria that are not directly supported by
simple voice recording, locating a specific recording may
require tedious and repetitive searching. For example, if
there is a need to find a specific customer's call to resolve a
disputed transaction, the recording unit or channel that
carried the original call might not be known, so the searcher
would be forced to manually play back many calls before
finding the correct one.

With the advent of computer telephony integration (CTI),
it is now possible to monitor a data link that supplies more
information about telephone calls, in addition to simple
voice recording. In a typical CTI system a telephone switch
or private branch exchange (PBX) provides an interface
suitable for processing by a computer, and expanded infor-
mation about telephone calls is made available through this
interface as the calls occur. Data fields that are available
within this expanded information may include the external
telephone number of the calling party, as well as identifi-
cation numbers to help associate a series of events pertaining
to the same call. With such a data link being used alongside
a voice recording system, the search and retrieval system can
be supplemented by constructing a database that combines the
previously discussed basic search criteria with enhanced
search criteria (based upon information obtained through a
CTI data link) such as: telephone numbers of parties
involved in the call; Caller ID (CLID) or Automatic Number
Identification (ANI); Dialed Number Identification Service
(DNIS); or the Agent ID Number of the Customer Service
Representative.

As shown in FIG. 2, with suitable equipment for tapping
into a voice communications line, a recording unit can
intercept telephone call traffic using two methods. By attach-
ing wires for recording channels on each extension within a
call center, the traffic can be intercepted and recorded as it
passes between the PBX and the agent telephone set. This
first method is known as "station-side" recording 180.
Alternatively, by attaching equipment on the trunk lines
between the PBX and Public Switched Telephone Network
(PSTN), the traffic can be intercepted at its point of entry

2

into the call center before the calls are dispatched by the
PBX. This second method is known as "trunk-side" record-
ing 170. Since businesses usually have more agent telephone
sets than trunk lines, a "trunk-side" solution is likely to
require less recording equipment and thus be less expensive.
Another significant point for consideration is that "trunk-
side" provides access only to external inbound or outbound
calls, which are those typically involving customers of a
business, whereas "station-side" also provides access to
internal calls between agents (which may or may not relate
to an external customer's transaction).

With respect to data links to provide call information to
computers, there are typically two different categories of
links from the PBX available. Some older links use inter-
faces such as SMDR (Station Message Detail Recording) or
CDR (Call Detail Recording) that provide summary infor-
mation about telephone calls in a line-oriented text format.
Both acronyms refer to essentially the same type of system.
Information from these links is generally provided after the
call has concluded, and as such is suitable for billing
applications or traffic analysis software. Many newer links
use real-time interfaces that are designed to supply a series
of events while a telephone call is still active within the
PBX, to enable computer and multimedia systems to
respond and interact with an external caller. The information
provided by such real-time links is typically much more
detailed than that provided by SMDR.

The detailed information and real-time nature of a CTI
link is particularly important when building a recording
system that is intended to react to telephone calls as they
occur and to dynamically select which calls ought to be
recorded or discarded. CTI-supplied information is also
important when building a recording system that is intended
to capture the full history of a telephone call, including
recording the different agents who were involved in the
conversation and how the call was held, transferred or
conferenced. Likewise, real-time information is important in
a system that intends to support (a) a live display of active
calls, and (b) the capability for a user to listen and monitor
the live audio traffic.

A "trunk-side" solution based upon voice recording alone
will not satisfy the above requirements in a practical manner,
since telephone calls are assigned to trunks dynamically as
needed to handle the traffic. What trunk channel a particular
call will be carried on cannot be predicted in advance.
Without information to associate a logical telephone call
with a physical recording of audio from a trunk channel, a
user might have to search and retrieve many recordings
before finding the one that is of interest. Moreover, in a
system designed to make use of the enhanced search criteria
provided by a data link, it would not be possible to pro-
grammatically associate the search data with the voice
recording without information about the trunk channel
where the call occurred.

This problem can be avoided as long as the data link
provides sufficient information about the trunk channels
being used for each call. Unfortunately, some PBX environ-
ments do not supply this critical information about trunk
channels within the data provided on the real-time CTI link.
For example, this problem is manifested by the Lucent
Technologies DEFINITY G3 PBX, which is a commonly
used telephone switch in North America. While the Lucent
G3 PBX provides trunk channel information through its
SMDR link, that information is not available until after the
conclusion of the call. This presents a problem for system
features and capabilities dependent upon real-time data. The
real-time data link provided by the Lucent G3 PBX does not

US 6,785,370 B2

3

provide the necessary information about trunk channels. There is thus a need for a system which is capable of simultaneously monitoring both the SMDR link and the real-time CTI link, gathering information about calls from both sources, and combining that information into a single data model of the telephony activity within the call center. There is a further need for a system that combines the data model with information concerning the location of call recordings, resulting in a "master call record" that contains data matching each call with the segments of which it is comprised, and matching the data for each segment with the location of the recording of that segment. Such a system would facilitate monitoring, recording, and playing back complete telephone calls.

SUMMARY OF THE INVENTION

The present invention is directed to a system and method that are capable of constructing a call record for each telephone call; receiving data regarding telephony events; matching a received telephony event with a call record; updating the matching call record based on the received telephony event data; and combining the updated call record with data indicating the location of recorded audio data for the segment of the call, to generate a master call record representing the lifetime of the telephone call. In further embodiments the system and method are capable of assembling and playing back segments of telephone calls using the recorder locations described in the master call record for each telephone call, and of using said master call record to display a graphical representation of said telephone call.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of the system of this invention in a preferred embodiment.

FIG. 2 illustrates the difference between trunk-side and station-side recording.

FIG. 3 shows a line-chart that illustrates various parties involved in a complex call.

FIG. 4 shows a schematic block diagram of a preferred embodiment for translating, summarizing and normalizing signals received from both an SMDR link and a Dialogic CT-Connect CTI service.

FIG. 5 illustrates the steps by which the translation module CtiCtc.exe integrates the data received from the CTI and SMDR links.

FIG. 6 illustrates how the CTI Server can be viewed as a set of logically distinct layers that deal with translating and distributing CTI events.

FIG. 7 illustrates how, in addition to telephony events, the CTI Server 710 is responsible for supplying certain metadata regarding agent events to the System Controller 130.

FIG. 8 shows the layout of the CTI Server.

FIG. 9 shows a version of CtiCtc.exe configured to work with a Lucent Telephony Services interface (and thus called CtiLts.exe instead of CtiCtc.exe).

FIG. 10 depicts key elements of the data model used in a preferred embodiment.

FIG. 11 illustrates three distinct layers of the CTI Server in a preferred embodiment.

FIG. 12 shows in block diagram form several threads of the CTI Server in a preferred embodiment that implement three distinct layers of processing (data collection, data normalization, and message emission).

FIG. 13 illustrates the program logic flow of the analyzer layer of the preferred embodiment.

4

FIG. 14 depicts the flow of information within the recording system of this invention in a preferred embodiment.

FIG. 15 shows how a recording unit operating with only voice signaling to guide the creation of its call records could make a number of fragmented audio segments.

FIG. 16 shows a graphical user interface used in the preferred embodiment.

FIG. 16A shows a system containing a CTI Server and a Recorder in a specific embodiment of the present invention.

FIG. 16B is a table illustrating descriptive information from the CTI Server used in a specific embodiment.

FIG. 17 illustrates steps in the creation of a Master Call Record used in a specific embodiment.

FIG. 18 shows the processing threads and data structures that comprise the CRG module in accordance with the present invention.

FIG. 19 illustrates the class diagram of the Call Record Generator used in a specific embodiment.

FIGS. 20, 20A, 20B, 21, 22, 22A, 22B, and 22C illustrate the operation of the Stream Control Manager.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The present invention is directed to a communication recording system and method. Generally, the functionality of the system involves tapping into activity on a PBX (Private Branch Exchange) by intercepting audio on either the trunk or station side of a telephone call. The tapped audio is then redirected as input to a channel on a Digital Signal Processor (DSP) based voice processing board, which in turn is digitized into program addressable buffers. The recorded digitized audio is then combined with descriptive information ("metadata") obtained through a Computer Telephony Integration (CTI) communications link with the PBX, and stored as a single manageable unit ("Voicedata") to facilitate its subsequent search and retrieval. The system uses modular architecture in both its hardware and software, so that any one component can be replaced or upgraded without affecting the rest of the system.

In a preferred embodiment the communications recording system comprises multiple rack-mountable computer-processing servers (such as the Compaq ProLiant 1600 R), using a multi-tasking operating system (e.g., Microsoft Windows NT), DSP voice processing boards (e.g., Dialogic D/160SC), and a distributed set of software components available from Dictaphone Corporation. In a specific embodiment directed to the smallest configuration, all of these components may reside in a single computer-processing server. In other preferred embodiments, related components are typically packaged in combinations and the entire system spans multiple servers that coordinate processing through a Local Area Network (LAN).

In this preferred configuration, the overall system generally comprises CTI Servers, Voice Servers, a Central Database Server, and User Workstations. CTI servers generally use a set of components to manage a data communications link with a telephone switch environment, to obtain notification of calls as they occur, along with the descriptive information about the calls (e.g., source and destination telephone numbers). The Voice Servers use a set of components to collect audio recordings, manage their storage, and facilitate their playback through the LAN. The Central Database Server uses a set of components to manage system-wide search and retrieval of recorded calls. User Workstations are typically desktop computers that use a set

US 6,785,370 B2

5

of components to allow a person to submit requests to search and retrieve recorded calls for playback and to control automatically scheduled functions within the recording system.

FIG. 1 shows in a block diagram from components of the system of this invention in a preferred embodiment. Data enters the recording system from a variety of sources. These sources can include a PBX 100, CTI middleware 105, ISDN lines 110, or other input sources 115. It will thus be appreciated that the system of the present invention can be used for monitoring and recording of information about any type of electronic communications. For simplicity, the following discussion uses the term Telephone calls. However, it is intended that term covers any electronic communication unless specified otherwise expressly.

Data from data sources 100, 105, 110 or 115 is transmitted to one or more CTI Translation Modules 165, which translates input data into a common format. The data is then sent to a CTI Message Router 120, which distributes the data onward to appropriate components of the system.

Audio Recorders 145 may be used for passive trunk-side 170 and extension-side 180 recording on a pre-determined static set of devices, as well as dynamically initiated recording of specific devices according to scheduling criteria through the Service Observance feature 185 provided by a telephone switch environment. The recordings are stored on an audio storage device 140. A Call Record Generator 150 matches data from the Audio Recorders 145 with data sent by the CTI Message Router 120 to create a Master Call Record (MCR) for each telephone call. The MCRs are stored in a Voicedata Storage module 155. One or more User Workstations 160 use the MCRs to reconstruct and play back complete or partial phone conversations stored in the audio storage device 140. A Scheduling and Control Services module 130 controls the Audio Recorders 145 and communicates with User Workstation 160. The Scheduling and Control Services module is responsible for starting and stopping the audio recording activity, according to pre-defined rules that are dependent upon time data provided by the Time Service 115 and CTI information. As the system components are packaged in the typical configuration, the CTI translation modules 165 and CTI message router 120 are co-resident upon a computer-processing server called the CTI Server 710. In a similar fashion, the combined set of components including the Time Service 125, Scheduling & Control Services 130, Audio Recorder 145, Audio Storage 140, and Call Record Generator 150, in a specific embodiment can be co-resident upon a computer-processing server called the Voice Server 124. The Voicedata storage 155 resides within a computer-processing server called the Central Database Server. The specialized application software for the User Workstation 160 resides upon desktop computers that use, in a preferred embodiment, Microsoft Windows 95, Windows 98 or Windows NT operating systems.

As noted above, in a specific embodiment the CTI Server comprises two main modules: a CTI translation module (such as the software program CtiCtc.exe, CtiLts.exe, and other translation modules) and a CTI Message Router module (such as the software program CtiServ.exe discussed below, or its equivalent). In a specific embodiment, the CTI Server may have several translation modules, for example, one for each PBX interface, or for each vendor API layer. As shown in FIG. 1, the CTI Server of the preferred embodiment accepts data from a PBX or similar equipment in a telephone switch environment, and can use both real-time CTI communications links and asynchronous information sources such the Station Message Detail Recording (SMDR)

6

interface. The CTI Server translates and combines the various types of input data into a unified, normalized format. Normalized format information is then passed by the Message Router to various components of the system, as required.

As noted above, the Voice Server in a specific embodiment has several modules, including the Audio Recorder 145 and Call Record Generator (CRG) 150. The Audio Recorder collects a plurality of audio segments, representing the portions of a telephone call during which the sound level exceeded an adjustable tolerance threshold, thereby discerning alternating periods of speech and silence. Functionally, the Call Record Generator (CRG) produces Master Call Records, which encapsulate information (metadata) describing a telephone call. This descriptive information comes from a plurality of sources, including but not limited to an Audio Recorder and a CTI Server. The call records are created using a participant-oriented Call Record Model. The CRG then attempts to match the call records with existing recorded audio data. The CRG is thus able to combine data arriving in different chronological order into a single manageable entity which describes the complete history of a telephone call.

In a specific embodiment, a Playback Server (PBServer) (not shown) is a sub-component within the Audio Recorder module which uses call records to retrieve and play back telephone calls. Each recorder has its own PBServer, which is connected to a Player module (not shown) on the User Workstation 160. The Player module generally contains a Stream Control Manager module, which enables the Player module to use the PBServers to play back a telephone call which has several different participants and thus may have portions of the call stored on different recorders.

CTI Server

Still with reference to FIG. 1, when a call comes into the PBX system, both SMDR and real-time CTI data are generated by the PBX, and supplied to the recording system via the SMDR and CTI links. In accordance with the present invention, these two types of data are integrated by the CTI Server into a common format.

As known in the art, CTI (Computer Telephony Integration) supplements the recorded audio data in several important ways. CTI data is provided through a data communications link from specific telephone switching equipment located at the customer site. Supplied data comprises such items as the telephone numbers of involved parties, caller ID/ANI information, DNIS information, and agent ID numbers. ANI is Automatic Number Identification, a signaling method that identifies the telephone number of the calling party; the method is typically used by large-scale commercial call centers. DNIS is Dialed Number Identification Service, a feature that identifies the original "dialed digits," and that is commonly used in large-scale commercial call centers when multiple directory numbers are routed to the same receiving trunk group. In accordance with the present invention, the CTI server performs the task of analyzing and reorganizing data from both the real-time (CTI) and SMDR (asynchronous) links, and passing the results onwards into the recording system for further processing.

The design of the system of the preferred embodiment envisions that there will be a number of CTI translation modules 165 to accommodate a variety of possible input sources such as "native" PBX interfaces, CTI "middleware" vendors, ISDN data channel interfaces, etc. The system

US 6,785,370 B2

7

design incorporates flexibility in the manner in which CTI information is collected, making the system prepared to integrate with CTI links that may already exist at a customer site. The CTI Server of the preferred embodiment is capable of simultaneously monitoring both an SMDR link and a real-time CTI link, gathering information about calls from both sources, and combining that information into a single data model of the telephony activity within the call center.

The CTI Server is responsible for supplying certain metadata regarding telephony events to the Voice Server's Call Record Generator 150. This metadata, such as called party and calling party numbers, trunk and channel ID, date and time, agent ID, etc., is combined by the Call Record Generator along with the other metadata, and data that is provided by the Audio Recorder 145 itself. Using this information, other components within the system are able to search for calls using a wide variety of useful and meaningful criteria, rather than simply using the recorder channel, date and time. As is known to those skilled in the art, an "event" is simply an action or occurrence detected by a computer program. The Call Record Generator 150 integrates that data into a single call record, which is updated after every event during the call, so that at the end of the call, the call's entire history is contained in the call record. The CRG matches the call record to the recording segments created by the Audio Recorders. The CRG integrates the call record with the metadata for the associated recordings of the phone call to generate a Master Call Record. When an operator wants to hear a recorded phone call, he uses the User Workstation (preferably equipped with a graphical user interface) to recall and play back the recorded call. Since the phone call may have had several different participants, pieces of the call may have been recorded on different recorders, each of which is associated with a different Playback Server. The system is nevertheless capable of playing back the entire phone call in the proper sequence.

In a preferred embodiment the CTI Server obtains the information regarding telephony events from various telephone switching environments, including PBXs, ACDs, and turret systems, which may have a wide variety of proprietary CTI interfaces. A telephone switching environment is a local telephone system that provides for routing of calls on a static or dynamic basis between specific destinations; the system is capable of identifying of when calls occur and who is involved in the calls. The CTI Server converts the information received into a common "normalized" format that is a simplified subset of the types of information available across the different vendors' PBXs, ACDs, and turret systems. This data conversion is partially facilitated by products such as Dialogic's CT-Connect API, which is capable of processing CTI messages from the major vendor's switches such as the Lucent DEFINITY G3, Nortel Meridian and DMS-100, Aspect, Rolm 9751, Rockwell Spectrum and Galaxy, Siemens Hicom, and Intecom. However, in accordance with the preferred embodiment an additional software layer exists within the CTI Server to further filter and normalize the CTI information. This feature also allows for a separate point of integration with customized software interfaces that may be necessary to connect with other switch vendors, especially certain turret systems that are not supported by Dialogic's CT-Connect (CTC) product. Alternate embodiments of the translation module use Lucent CentreVu Computer Telephony Server for Windows NT, or Genesys T-Server, as middleware instead of Dialogic CT-Connect. Additional alternate embodiments include direct "native" interfaces to a particular telephone switch, such as Aspect, without an interposing middleware product.

8

In terms of the CTI messages exchanged between the CTI Server and the various PBXs, ACDs, and turret systems, in accordance with a preferred embodiment the CTI Server is a "passive listener." That is, the CTI Server will monitor and receive information about call activity, but it will not send messages to affect, control, or redirect the calls. Using an "active" CTI server is also contemplated in specific embodiments.

Whereas the focal point of a Voice Server is recording content (e.g., audio clips), the metadata generated by the CTI Server is focused on describing the facts pertinent to the start and end points of each participant's involvement within a call. In other words, within the system of the preferred embodiment, recording is managed in a call-centric (rather than event-centric) fashion. This corresponds with the typical caller's point of view, in which a call is the entire conversation with a business entity, even if the conversation involves transfers to other agents or conferencing of multiple parties. The CTI Server generates events with metadata for the start and end points of the various recording segments of a complex conversation. These event records are interrelated by ID numbers and reason codes (see FIG. 3) so that the entire sequence of events for a complex conversation can be reconstructed by a browser application, preferably implemented on the User Workstation 160.

In accordance with the preferred embodiment, there can be one or more CTI Servers within the system of the subject system, as needed to process the traffic load of CTI information generated by multiple PBXs, ACDs, and turret systems. In a specific embodiment, a single CTI Server may be configured so as to connect with several PBXs, ACDs, and turret systems, depending upon the traffic load and physical connectivity requirements. In alternate embodiments, different CTI servers can be attached to different input sources. Generally, the number of CTI Servers within the system does not have a direct relationship with the number of Voice Servers. The telephony events generated by a CTI Server are individually filtered and re-transmitted to the appropriate Voice Server based upon configuration data for the system as a whole (managed by the Central Database Server), which maps the recording locations (extension number, or trunk & channel ID) with the Voice Server name and recording input port (channel).

During the active lifetime of a call, real-time information is accumulated within a historical call record that tracks each participant within the call. Each participant record includes descriptive fields for telephone numbers, agent ID numbers, time ranges, and reason codes for joining and leaving the conversation. At certain key points during the accumulation of data, whenever a party joins or leaves the conversation, the call record is transmitted onward to allow the rest of the recording system to process the information accumulated thus far. Upon the conclusion of the call, the CTI server retains a copy of the call record for a configurable time interval before discarding it from memory. This delay is intended to allow for the arrival of the SMDR data.

Upon receiving SMDR data, the CTI server searches its memory for a call record pertaining to the same logical telephone call that would have been accumulated from previous real-time messages. Matching this information is not a trivial task, since the SMDR link and real-time CTI link do not share a common reference ID number for use by their messages in describing the occurrence of the telephone call.

Therefore the software of the preferred embodiment must use other "clues" to guide the matching process, by com-

US 6,785,370 B2

9

parison on a combination of other data fields that exists in common between the SMDR and real-time CTI data. These data fields include: (1) the telephone number of the first external party involved in the call; (2) the telephone number of the first internal party involved in the call; (3) the direction of the call (e.g., inbound, outbound); (4) the start time of the call, in hours and minutes; and (5) the duration, in seconds, of the call.

Once again, the matching process is not trivial because the SMDR link gives the starting time of the call only in hours and minutes, whereas the starting time given by the real-time link also includes seconds. It is quite possible that more than one call could be started and stopped within a single minute. This would result in an ambiguous match, if not combined with other search fields. The same argument holds true for each of the other fields upon which a match can be performed. No single field alone will provide an unambiguous matching of the records. Even in combination, it is conceivable (although statistically unlikely) that an ambiguous case could occur: if the same two parties were to call each other twice within the span of a minute, and each call was roughly the same length in seconds. The odds of such a problem are increased if a large number of calls are routed through a common entry point into the call center, as would be the case if the first internal party involved in the call is a shared Voice Response Unit (VRU) or Automatic Call Distribution (ACD) queue. In addition, if information about the external party's number is missing due to limitations of the PSTN or incoming trunk configuration, matching the call records becomes even more problematic.

Adding to these difficulties is the fact that clock-time values reported by the SMDR link and the real-time CTI link may not be perfectly in synchronization with each other. Therefore, the preferred embodiment comprises a mechanism in which an imperfect match of times can be tolerated, while still retaining an acceptable level of reliability in matching the call records.

Because these various factors require a degree of flexibility in the matching algorithm, the preferred embodiment incorporates a weighted formula that is applied to potential match candidates. The formula yields a numerical confidence factor that can be used to select the best apparent match candidate. For each of the "clues," a test is conducted to determine the quality of matching on that data field. This matching quality is rated as a percentage. Certain fields, such as time values, are allowed to vary within a configurable tolerance range, whereas other fields are required to match exactly or not at all. After the matching quality of a field has been determined, it is multiplied by an importance factor that applies a relative weight to each of the various fields that can be examined during matching. The final confidence factor is the summation of these calculations:

Confidence Factor= $\sum_i (Match\ Quality)_i * (Weighting\ Factor)_i$

In order to account for the fact that characteristics of the call traffic may vary significantly between individual call centers, the tolerance factors (e.g., for time value offsets) and the weighting factors are re-configurable. There is also a re-configurable minimum level for confidence factors, below which the match candidate will always be rejected.

For those fields, such as time or duration, where an imprecise match may be allowed, the configuration data will define an allowable variance range (plus or minus a certain number of seconds). Values that do not match exactly, but fall within the variance range, are rated with match quality expressed in percentage that is measured by one minus the

10

ratio of the difference from the expected value versus the maximum variance.

Match Quality= $1 - (abs(Expected\ Value - Actual\ Value) / Maximum\ Variance)$

Values outside the variance range are rated as a match quality of zero. This produces a linearly scaled match quality. Alternate embodiments may use other distributions (e.g., standard deviation "bell curves") to produce a non-linear scale for the match quality. Where an exact match is required for a field, the match quality is either 100% or zero.

EXAMPLE

Real-time CTI events report a telephone call from an unknown external party (missing or deliberately suppressed ANI/CLID information) to an internal party at extension 1234, starting at 12:25:03 and lasting for 17 seconds (CLID is Calling Line Identification, a signaling method that identifies the telephone number of the calling party; the method is typically used by residential subscribers and small businesses). Two SMDR records arrive which could possibly match with this call. The first record indicates an inbound call received by extension 1234 at 12:26 and lasting 26 seconds. The second record indicates an inbound call received by extension 1234 at 12:27 and lasting 20 seconds. The system is configured with a variance range of plus or minus 3 minutes for the start time, and plus or minus 10 seconds for the duration.

Weighting Factors are:	
20	External Party Telephone Number
40	Internal Party Telephone Number
30	Direction
20	Start Time
20	Duration

Confidence Factors are therefore calculated as follows:

$$CF_1 = (20 * 1.00) + (40 * 1.00) + (30 * 1.00) + (20 * (1 - 1/3)) + (20 * (1 - 9/10))$$
$$= 105\ 1/3$$
$$CF_2 = (20 * 1.00) + (40 * 1.00) + (30 * 1.00) + (20 * (1 - 2/3)) + (20 * (1 - 3/10))$$
$$= 110\ 2/3$$

The system will therefore match the CTI events with the second SMDR record.

After a match has been selected, the trunk channel information (and any other useful information that can supplement the previously gathered real-time CTI data) is extracted from the SMDR data and added to the call record within the CTI server's data model of telephony activity. Then the updated call record is transmitted onward to allow the rest of the recording system to process it. With the trunk channel information at hand, the recording system is able to associate the enhanced logical search information with the physical voice recording, and take whatever actions may have been dependent upon this information, such as selectively recording or discarding the call.

FIG. 2 is an illustration of the difference between trunk-side and station-side recording at a call center with agents. With suitable equipment for tapping into a voice communications line, a recording unit can intercept telephone call

US 6,785,370 B2

11

traffic using either of these two methods. By attaching wires for recording channels 180 on each extension within a call center, the traffic can be intercepted and recorded as it passes between the PBX 100 and the agent telephone sets 230. This first method is known as "station-side" recording. Alternatively, by attaching equipment 170 on the trunk lines between the PBX and Public Switched Telephone Network (PSTN) 250, the traffic can be intercepted at its point of entry into the call center before the calls are dispatched by the PBX. This second method is known as "trunk-side" recording. Since businesses usually have more agent telephone sets than trunk lines, a "trunk-side" solution is likely to require less recording equipment and thus be less expensive. Another significant point for consideration is that "trunk-side" provides access only to external inbound or outbound calls, which are those typically involving customers of a business, whereas "station-side" also provides access to internal calls between agents (which may or may not relate to an external customer's transaction).

A third type of recording interface is Service Observance 185 (see FIG. 1), which is physically wired in manner like station-side recording, but using separated dedicated lines to a recording input channel rather than being interposed between a PBX and telephone set. In this mode of operation, the Recorder joins into a telephone call as a silent conference participant using the PBX Service Observance feature (originally intended to enable a supervisor to directly monitor an employee's telephone calls upon demand). This differs from ordinary station-side recording in that the internal party being recorded on a given input channel can vary upon demand rather than being fixed by the wiring pattern.

FIG. 3 shows a line-chart that illustrates various parties involved in a complex call. A is the customer phone number, and B and C are the agent phone numbers located behind recording channels R20 and R21 respectively (see FIG. 2).

Initially, the call comes in from line A 335 to line B 340. A real-time CTI message occurs describing that phone B is ringing, but not yet answered. B answers the phone 365 at time t0 310. The "NS" at 360 indicates the normal start of a phone call. A real-time CTI message occurs describing the start of the call between A and B. The telephony model is updated to reflect the fact that the call between the initial 2 participants (A and B) started normally at time t0 310. A copy of the call record is then sent onward to the rest of the recording system. The call record is retained within the telephony model, associated with device (or line) B. At time t1 315, B places the call on hold 370 (the "XA" at 370 indicates that the call was transferred away from B; the "XR" at 375 indicates that the transfer was received by HOLD). A real time CTI message occurs describing that B placed the call on hold. The telephony model is updated to reflect that B transferred the call to HOLD 345 at time t1 315. (This information is accumulated with the information previously gathered at t0 310). A copy of the call record is then sent onward to the rest of the recording system. The call record is removed from device B within the telephony model, but kept in a list of held calls.

At time t2 320, B returns to the call 380 and conferences in C 355 (the "XA" at 380 indicates that the call was transferred away from HOLD; the "XR" at 382 indicates that the transfer was received by B; the "CA" at 384 indicates that C was added as a conference participant). A real-time CTI message occurs describing that B returned to the call and invited C by conferencing. The call record is moved within the telephony model from the list of held calls back to device B. The telephony model is updated to reflect

12

that HOLD 345 transferred the call 380 back to B at t2 320. (Note that information is accumulated with the information previously gathered at t0 310 and t1 315). A copy of the call record is then sent onward to the rest of the recording system. The telephony model is updated to reflect that C joined the call 384 as a conference participant at t2. (This information continues to be accumulated with previously gathered information). A copy of the call record is then sent onward to the rest of the recording system. The call record is retained with both devices B and C within the telephony model.

At time t3 325, a real-time CTI message occurs describing that C dropped out 386 of the call (the "CD" at 386 indicates that C was dropped from the conference). The telephony model is updated to reflect that C dropped out of the conference at t3. (This information continues to be accumulated with previously gathered information). A copy of the call record is sent onward to the rest of the recording system. The call record is removed from device C within the telephony model, but retained with device B.

At time t4 330, A terminates the call to B. A real-time CTI message occurs describing that A terminated the call (The "ND" at 390 indicates that a normal drop of the call occurred; the "OPH" at 395 indicates that the other party hung up). The telephony model is updated to reflect that A stopped normally and B stopped because the other party hung up at t4. (This information continues to be accumulated with previously gathered information). A copy of the call record is then sent onward to the rest of the recording system. The call record is then removed from device B, but kept in a list of completed calls. An SMDR message is received which summarizes the call in its entirety. The list of completed calls is searched to find a match, and the appropriate call record is retrieved. The call record is updated with the trunk channel information from the SMDR message. A copy of the call record is sent onward to the rest of the recording system. The call record is removed from the list of completed calls.

FIG. 4 shows a schematic block diagram of a preferred embodiment for translating, summarizing and normalizing signals received from both an SMDR link and a Dialogic CT-Connect CTI unit. In the embodiment illustrated in FIG. 4, the recording system of the subject system is represented by daVinci™, a new generation recording system of Dictaphone Corp. Alternatively (or simultaneously), Dictaphone's Symphony™ CTI software can be used, in conjunction with Dictaphone's ProLog™ recording system (the system preceding daVinci). Hereinafter, the translation/summarization module of the preferred embodiment illustrated in FIG. 4 will be referred to as CtiCtc.exe.

The module CtiCtc.exe is itself comprised of a plurality of modules, as shown in FIG. 4. A CtiAgentEvent module 448 is comprised of a data structure for agent log-on and log-off messages. A CtiAgentStatusFile module 454 manages a file that tracks agents currently logged on. A CtiCallEvent module 416 is comprised of a data structure for a call record (i.e., normalized and summarized CTI events). A CtiCall-State module 418 is comprised of a generic data structure to represent the state of telephony activity at a particular location (extension, hold area, etc.). A CtiComMessageEmitter module 476 comprises a layer that converts the stream of CtiCallEvent objects (generated by a CtiCtcAnalyzer 456) into a format that can be sent to other da Vinci system components. A CtiCtcAnalyzer module 456 comprises a processing engine which examines CTC and SMDR messages and keeps track of a state machine for the activity on each extension. The CtiCtcAnalyzer module performs normalization of the CTC and SMDR data.

US 6,785,370 B2

13

A CtiCtcAnalyzerUtils module 452 comprises a collection of utility subroutines that assist in examining the CTC and SMDR messages. A CtiCtcCallState module 420 comprises a data structure that represents the state of telephony activity at a particular location (extension, hold area, etc.) including CTC-specific information. A CtiCtcCallStateList module 432 manages an open-ended collection of CtiCtcCallState objects. This collection of objects is typically used to track calls that are "held" or "bumped." A CtiCtcData module 428 comprises a data structure wrapped around the raw CTC data, with the addition of a time stamp indicating when a message arrives. A CtiCtcDataFile module 412 manages a file of CtiCtcData objects that can be captured or displayed. A CtiCtcExtensionInfo module 442 manages a collection of CtiCtcCallState objects, with one object for each extension.

A CtiCtcInput module 464 comprises an input source engine that obtains incoming CtiCtcData objects, either from a "live" server or from a playback file. A CtiCtcMain module comprises the main() function for CtiCtc.exe. The main() function handles command line and registry parameters, along with other start-up processing. A CtiCtcParameters module 472 comprises data structure and program logic for managing the configuration parameters in the Windows NT registry. A CtiCtcScanner module 446 comprises a utility module for building a list of all available extensions on a particular telephone switch. A CtiCtcStats module 434 comprises a data structure for compiling statistics on the number of CTC, SMDR, and CTI messages. A CtiDtpField module (not shown) is used by a CtiDtpMessageEmitter module 478, and comprises a data structure for an individual field in the Dictaphone Telephony Protocol ("DTP"), used to communicate with other Symphony CTI system components. A CtiDtpMessage module (not shown) is used by a CtiDtpMessageEmitter module 478, and comprises a data structure for a complete message in the DTP to be sent onwards to the Symphony CTI system.

A CtiDtpMessageEmitter module 478 comprises a layer that converts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) into a format that can be sent to the Symphony CTI recording platform. A CtiDtpSocketSrv module (not shown) manages the TCP/IP connection through which messages for DTP are sent to the Symphony CTI platform. A CtiDtpUtility module (not shown) comprises a collection of utility routines that assist in examining and processing DTP messages. A CtiExtensionFile module 450 manages the configuration file that lists all available telephone extensions. A CtiExtensionInfo module 440 manages a collection of CtiCallState objects, with one object for each extension. A CtiExtensionNumber module 430 comprises an abstraction of an individual extension number as either a numerical or string value, so that changes to this model will not have a global impact in CtiCtc.exe.

A CtiMessageEmitter module 458 comprises an abstract layer that converts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) into a format that can be sent to various target platforms, including the da Vinci and SymphonyCTI systems. A CtiMessageEmitterParameters module 474 comprises a data structure and program logic for managing configuration parameters that relate only to the message emitter(s). A CtiMessageQueue module 462 comprises shared memory for transferring data between threads. As is known to those skilled in the art, a "thread" is a part of a program that can execute independently of other parts. A CtiNullMessageEmitter module 460 comprises a layer that accepts the stream of CtiCallEvent objects (generated by CtiCtcAnalyzer 456) and discards them instead of sending

14

them to a target platform. Typically this layer is used only when debugging CtiCtc.exe, or to capture a sample file of CTI events from a PBX without sending them to the da Vinci or SymphonyCTI systems. A CtiPartyListElement module 414 comprises a sub-component of the CtiCallEvent data structure 416. The module 414 tracks information about an individual participant (e.g., caller, recipient) in a call.

A CtiPeriodicMsg module 468 comprises a generic handler for sending timer-based housekeeping messages. A CtiPrint module 444 comprises a layer that manages console output and conditional trace messages. A CtiSmdrData module 424 comprises a data structure wrapped around the raw SMDR data, with the addition of a time stamp indicating when a message arrives. A CtiSmdrDataFile module 408 manages a file of CtiSmdrData objects that can be captured or replayed. A CtiSmdrDataList module 422 manages an open-ended collection of CtiSmdrData objects. This is typically used to buffer SMDR records that have not been paired with CTC records. A CtiSmdrInputModule 466 comprises an input source engine that obtains incoming CtiSmdrData objects, either from a "live" server or from a playback file.

A CtiTagNames module 436 comprises a utility module that converts number values to descriptive strings for debugging and tracing purposes. A CtiTime module 438 comprises a utility module that converts time values to UTC for internal storage and conditionally prints times in either the UTC or local time zone. A CtiTrunkMap module 426 comprises a data structure that describes a mapping between logical trunks and logical trunk groups, into physical trunks and TDM timeslots. A CtiTrunkMapFile module 410 manages a configuration file that contains the CtiTrunkMap information.

FIG. 5 illustrates the steps by which the translation module CtiCtc.exe integrates the data received from the CTI and SMDR links. Initially, at step 502, the translation module receives a message from the SMDR link or from the CTI link. If the message is determined, at step 504, to be a CTI message, the current data model of telephony activity is updated at step 506. If the translation module determines at step 514 that the CTI message indicates a party joined or left the call, the call record is at step 518 transmitted onward to the rest of the recording system before continuing to step 512. Otherwise, no message is transmitted onward to the rest of the recording system and processing continues directly to step 512. If the translation module determines at step 512 that the CTI message indicates that the call has been concluded, at step 520 the module removes the call record from the associated devices. The translation module then adds the call record to the list of recently completed calls at step 528. Completed calls are discarded (step 530) after they get too old (i.e., after a predetermined number of recorded calls, or a given time period after the original recording of the call). Processing then continues again from step 502 by receiving the next incoming message. If at step 512 the call has not been concluded, the completed calls are discarded (step 530) after they get too old. Processing then continues again from step 502 by receiving the next incoming message.

If at step 504 the message is an SMDR message, the translation module at step 508 scans the list of recently completed calls. At step 510 the translation module calculates confidence factors for the recently completed calls by using the formula:

$$\text{Confidence Factor} = \Sigma_i (\text{Match Quality}_i) * (\text{Weighting Factor}_i)$$

If any matches are found (step 516), and more than one match is found (step 522), the match with the highest

US 6,785,370 B2

15

confidence factor is used (step 526). If only one match is found, that match is used (step 524). At step 540, the trunk channel information is extracted, and at step 544 the call record is updated within the list of recently completed calls. The call record is transmitted at step 548 to the rest of the recording system. At step 550 the call record is discarded from the list of recently completed calls. Completed call are discarded (step 530) after they get too old. If no matches were found at step 516, the completed calls are discarded (step 530) after they get too old. Processing then continues again from step 502 by receiving the next incoming message.

As shown in FIG. 6, the CTI Server can be viewed as a set of logically distinct layers that deal with translating and distributing CTI events. Starting from the bottom of the picture, CTI events flow from a PBX in its proprietary format to Dialogic CT-Connect middleware 640 another API layer 650 or custom interface layer 660 that each provide partial normalization of the data. This helps to reduce the complexity of the "translation" job, since there are fewer APIs than individual PBX types. But since one object of the subject system is to retain the flexibility to integrate with a variety of third-party CTI vendors (e.g., Dialogic, Genesys, etc.) there is another layer 670 above the API or custom interface layer to complete the job of "translation." The final result after passing through this "normalization" layer is that all of the CTI events are in a single, common, integrated data format.

Once the CTI events have been converted to a normalized format, the CTI Server can address its other mission of distributing (routing) the messages. The distribution layer 680 examines each message to determine what other recording system components need to receive it, and then sends a copy of the event to the appropriate destination(s).

This logical separation of responsibilities used in a preferred embodiment simplifies the programming required to implement the subject system. Translation modules do not need to know anything about other recording system components, and they can focus on dealing with a single specific PBX or vendor API layer. Likewise, the distribution module will not need to know anything about specific PBX or vendor API layers, and it can focus on making routing decisions and communicating with the rest of the recording system.

FIG. 7 illustrates how, in addition to telephony events, the CTI Server 710 used in accordance with the present invention is responsible for supplying certain metadata regarding agent events to the System Controller, which is part of the Scheduling & Control Services 130 shown in FIG. 1. This information, which generally includes agent ID, extension number, logon and logoff time, etc., is obtained when available from the various PBXs, ACDs, and turret systems. The agent events delivered to the System Controller 130 enable a map to be maintained of the extension number(s) where a real person can be found, at a given date and time. This information enables a browser application to intelligently associate some of the previously recorded calls even if a person was using different telephone sets according to a 'free seating' plan. The CTI Server 710 also keeps a local cache of the agent information, so that agent information can be included when sending the telephony events to the Call Record Generator 150.

The physical layout of the CTI Server used in a specific embodiment is shown in FIG. 8. With reference to FIG. 1, the translation modules are implemented by separate programs, such as CtiCtc.exe 406, which encapsulate the details on converting a specific PBX interface or vendor API

16

layer into a normalized format. The distribution module is preferably implemented by a single program, CtiServ.exe 820, which includes the main processing and routing logic for the CTI Server.

As noted, the translation modules of the CTI Server convert proprietary-format CTI information into a normalized format. In accordance with a preferred embodiment, this is done in several layers within the program. The information is first converted by Dialogic's CT-Connect software into the CTC-API format, and then the conversion to the generic format used by the other components of the recording system is completed by the translation module CtiCtc.exe. Once the data is converted, it is transmitted to the distribution module (CtiServ.exe) by using a distributed communications method such as DCOM. Component Object Model (COM) is a Microsoft specification that defines the interaction between objects in the Windows environment. DCOM (Distributed Component Object Model) is the network version of COM that allows objects running on different computers attached to a network to interact. An alternate embodiment of the CTI Server utilized Microsoft Message Queue (MSMQ) technology as the means to carry messages among the system components, instead of the original DCOM method used by CtiServ.exe, and those skilled in the art would appreciate that a variety of additional data communications technologies are also suitable to this role.

The translation module and the distribution module of the CTI Server can be located on different machines, if desired. There can be multiple translation modules running in the system—one for each PBX or CTI middleware environment. There can also be different types of translation modules, with one version for each interface or API layer. As depicted in FIG. 8, CtiCtc.exe deals with the Dialogic CT-Connect API, and there are 3 copies of this program running to handle the PBXs. If other types of APIs are used, there would be other programs for these various interfaces. All translation modules contribute data upward to the distribution module in a single, common, normalized format. An example of a version of CtiCtc.exe configured to work with a Lucent Telephony Services interface (and thus called CtiLts.exe instead of CtiCtc.exe) is shown in FIG. 9. The modules which are common to both versions of the program are shown in FIG. 9 as shaded gray. The unshaded modules represent those portions of the program that necessarily vary between CtiCtc.exe and CtiLts.exe, due to the differing input parameters and data structures used by both systems.

Again with reference to FIG. 8, the distribution module (CtiServ.exe) receives and collects all the CTI events from the various translation modules. Then it puts the events into a single inbound queue 830 for processing by a main control thread 835. After the events are processed, they are separated into individual outbound queues 840. Finally, the events are sent by various delivery threads 850 to the CRG components within different Voice Servers. The main processing thread 855 (WinMain) is deliberately isolated (decoupled) from the inputs and outputs to ensure that delays in transmitting or receiving data will not impact the overall performance of the CTI Server.

FIG. 11 shows how the CTI server in accordance with a specific embodiment consists of several threads that implement three distinct layers of processing (data collection 1110, data normalization 1120, and message emission 1130). FIG. 12 illustrates the processing steps of these layers. The dashed lines indicate message flow between threads, whereas the solid lines indicate program logic flow. The CTI translation modules are thus internally separated into 3

US 6,785,370 B2

17

major sub-tasks: (1) data collection from the input source (PBX, CTI middleware, etc.); (2) normalization of the data to a common format; and (3) communications with the system platform.

In a data collection layer, the initial step 1210 is to open the connection to the CTI data source. At step 1214 the layer receives a CTI event, and at step 1216 posts the CTI event to the Message Queue 462 (see FIG. 4). If at step 1218 a shutdown is in progress, the connection to the CTI data source is closed at step 1220, and at step 1222 data collection is ended. If at step 1218 a shutdown is not in progress, the CTI connection remains open (step 1212).

At step 1228, the data normalization layer receives a CTI event from the Message Queue 462. The data normalization layer updates the telephony model at step 1230. See FIG. 13 for a more detailed explanation of the updating of the telephony model. At step 1231, the call state is posted to the Message Queue, if necessary. At step 1232 completed calls are discarded from memory after they age beyond a configurable time limit. At step 1233 the "hang-up" routine is called to update the telephony model for held or bumped calls after they age beyond a configurable time limit. At step 1234, if a shutdown is in progress, the data normalization layer checks the inbound message queue at step 1236. If the message queue is empty, data normalization is ended (step 1238). If the message queue is not empty at step 1236 or if there is not a shutdown in progress at step 1234, the data normalization layer goes to step 1226 and waits for the next CTI event to arrive.

18

The message emission process begins with opening a connection to a target platform, such as the da Vinci or SymphonyCTI recording systems at step 1240. At step 1244, the message emission layer receives the call state from the message queue 462. At step 1246, the call state data is converted into a platform-specific format. At step 1248, the message emitter sends the message to the target platform. At step 1250, if a shutdown is in progress, a check is made at step 1252 for whether the inbound message queue is empty. If the inbound message queue is empty, message emission is ended at step 1254. If the inbound message queue is not empty at step 1252, or if there is not a shutdown in progress at step 1250, the message emission layer, at step 1242, maintains the open connection to the target platform and awaits the next call state transmission.

Master Call Record

The CTI Server sends "Call Event Records" onward to the recording platform. These messages provide details on the start and end of calls, as well as significant transitions that affect the lists of participants for the calls. The list of participants is cumulative, and information regarding participants is retained for the entire duration of the call even when some participants in the list may have dropped off from the call. If a participant rejoins the call, a new, separate entry will be created to reflect that change within the participant list. The following table shows the fields contained within these messages.

CtiCallEvent		
Name	Type (max length)	Description
Version	WORD	Version number of this message format, for reverse compatibility.
MessageID	GUID	Unique ID for this message instance
RecorderNode	WORD	A number that identifies a particular Voice Server
RecorderChannel	WORD	A number that identifies a recording input channel on a Voice Server
EventType	BYTE	Indicates if this event added (0x01) and/or dropped (0x02) participants in the call.
EventReason	BYTE	Indicates if this call was affected by a normal (1), conference (2), or transfer (3) telephony event
CTICallRecId	GUID	Unique ID pertaining to entire call (CTI server provides the same ID for a call that is transferred, conferenced, etc)
CallDirection	BYTE	Indicates call origin - outbound (0x12), inbound (0x21), internal (0x11), or unknown (0x44)
RingLength	WORD	Seconds between the first ring signal and going off-hook (picking up the phone)
DTMFCode	String*(50)	DTMF codes entered during the call
ApplicationData	String*(32)	Character array dedicated to information the switch may provide along with the call (e.g., account number)
CallingParty	WORD	Index number of the calling party within the participant list. Normally this is zero.
CalledParty	WORD	Index number of the called party within the participant list. Normally this is one.
PBXCallRecId	DWORD	Number provided by the PBX to identify this call.
NumberOfParticipants	WORD	Count of participants in the following array.
ParticipantList	Vector*	Array of PartyListElement describing all participants involved in the call

*ObjectSpace data types

US 6,785,370 B2

19

ObjectSpace is a set of C++ class libraries provided by ObjectSpace, Inc., that define useful general-purpose data structures including representations of strings, time values, and collections of objects (such as vector arrays or linked

20

lists). These class libraries are implemented in a way that supports a wide variety of computer operating systems. Those skilled in the art will appreciate that many alternate implementations for such data structures are suitable for this role.

CtiPartyListElement		
Name	Type	Description
AgentID	String*(24)	Registered ID of person, typically used for "free seating" call center environments
Number	String*(24)	Telephone number of this participant (e.g., ANI, DNIS, Dialed Digits)
Console	String*(10)	Seating position that can consist of one or more stations.
Station	String*(10)	Unique telephone set, possibly with multiple extensions.
Extension	String*(6)	Internal line number of the participant
SwitchId	WORD	Number of the switch (PBX, ACD, or turret system) which is handling the conversation
TrunkID	WORD	Identification of trunk line which is handling the conversation
VirtChannel	WORD	Identification of trunk's channel (time slot) which is handling the conversation
LocationReference	BYTE	Describes the location of participant with respect to the switch - can be internal (1), external (2), or unknown (3)
StartTime	time_and_date*	Time participant joined the call
EndTime	time_and_date*	Time participant left the call
ConnectReason	BYTE	How participant joined the call: norm start of call (1), being added to a conference (2), or receiving a transferred call (3)
DisconnectReason	BYTE	How participant left the call: normal end of the call by hanging up (1), dropping out of a conference (2), transferring away a call (3), or call ends by another party hanging up (4).
Changed	BOOL	Indicates if recent change in CTI message.

*ObjectSpace data types

For external participants, only the fields Number, SwitchName, TrunkID, VirtChannel, LocationReference, StartTime, EndTime, ConnectReason, and DisconnectReason will be applicable. For internal participants, all fields may be applicable. Unused string fields will be null terminated. Unused number fields are set to zero. Each call event record will contain at least two participants in the list. These two participants are the original calling party (0) and called party (1) and will appear within the list in that order respectively.

Note: The data field "Number" will be filled in a variety of ways, depending upon the type of participant and direction of the call.

US 6,785,370 B2

21

Participant Type	Call Direction	Number Field
External participant	Inbound Call	ANI
External participant	Outbound Call	Dialed Digits
Internal participant	Inbound Call	DNIS or Extension
Internal participant	Outbound Call	Extension
Internal participant	Internal Call	Dialed Digits or Extension

The CTI Server sends "Agent Event Records" onward to the recording platform's System Controller to convey information when an agent logs on/off at a particular location. The following table shows the fields contained within these messages.

<u>CtiAgentEvent</u>		
Name	Type (max length)	Description
Version	WORD	Version number of this message format, for reverse compatibility.
MessageID	GUID	Unique ID for this message instance
EventType	BYTE	Indicates if this event pertains to either a logon (1) or logoff (2).
LocationType	BYTE	Indicates if this event pertains to a location type such as a console (1), station (2) or extension (3).
AgentID	String*(24)	Registered ID of person, typically used for "free seating" call center environments
SwitchId	WORD	Number of the switch (PBX, ACD, or turret system) where the agent connected.
Console	String*(10)	Seating position that can consist of one or more stations.
Station	String*(10)	Unique telephone set, possibly with multiple extensions.
Extension	String*(6)	Internal line number of the participant
StartTime	time_and_date*	Time that the agent logged in.
EndTime	time_and_date*	Time that the agent logged out.

*ObjectSpace data types

Within any given "Agent Event Record", only one of the following three fields will be applicable: Console, Station, or Extension. The actual mapping is determined by the LocationType. Unused string fields will be null terminated. Unused number fields are set to zero.

It will be appreciated that the general principles behind the method described above are suitable not only for associating and combining real-time CTI data with the trunk channel information from an SMDR message, but also for any situation where a mixture of information is being provided from two or more sources and there is a need to gather and merge the information to get a more complete picture of what is actually happening in the system. The disclosed method could easily be adapted by those of ordinary skill in the art to situations in which the mapping or association between the multiple sources of information is "weak" and prone to ambiguity. While this method does not make the potential ambiguity disappear, it helps to define a quantitative set of rules for making a judgement call on when a match is "good enough" to act upon. While human beings are often capable of making such judgement calls intuitively, computers need a specific set of instructions in order to act in a repeatable and reliable fashion upon the input data.

Previous recording systems that made use of CTI to collect enhanced search information mimicked the event-

22

oriented interfaces provided on the data links from a PBX. Individual database records were constructed on a 1-to-1 basis for the events occurring during the total lifetime of a phone call. The interpretation of the series of events was left to the end user. Associations between related events were made difficult in certain cases because the call identification numbers given by a PBX may change after a call has been transferred or conferenced, or the numbers may be recycled and reused over time. Following and tracing the history of events for a complete call from the perspective of the external customer could require much manual and repetitive searching. Playing back the entire set of audio recordings from the start of that customer's interaction with the business, to the ultimate conclusion of that customer's transaction, could also require additional repetitive manual

requests to play back the individual recorded segments within a call that was transferred or conferenced.

To resolve this problem, the CTI server of the preferred embodiment maintains and accumulates information within a data model of telephony activity. FIG. 10 depicts the key elements of the data model. This consolidated information is shared with the rest of the recording system when parties join or leave a call, thereby eliminating the need for downstream components to store or interpret the individual CTI events occurring during a call's lifetime.

During the active lifetime of a call, real-time information is accumulated within a historical call record that tracks each participant within the call. At certain key points during the accumulation of data, whenever a party joins or leaves the conversation, the call record is transmitted onward to allow the rest of the recording system to process the information accumulated to that point. Upon the conclusion of the call, the CTI server of the preferred embodiment retains a copy of the call record for a configurable time interval before discarding it from memory. This delay allows for the arrival of the SMDR data.

The call records are organized into a two-tiered hierarchy of calls and participants. Certain data fields that apply globally to the entire call are stored at the upper level. Most data fields, however, apply only to a specific party involved

US 6,785,370 B2

23

within a call, and are stored at the lower level. Individual participants can have identifying information (such as extension number, agent ID, telephone number via DNIS/ANI/CLID, trunk and channel) along with time-stamps and reason codes for the entry and exit from participation in the telephone call. Reason codes include initial start, transfer, hold, resume, conference add/drop, and hang-up.

The currently active call on each telephone set being monitored is maintained within a storage area 1020 of the data model. Also, the data model provides for an open-ended list 1040 of calls that may be "on hold" (and therefore not associated with any telephone set). There is also a list 1030 that can be used temporarily for calls when they are in a state of transition during transfers, queuing or re-routing, for the brief period of time when an active call is disassociated from its original telephone set but not yet associated with a new telephone set. Finally, there is a list 1050 of recently completed calls that is used to await additional information that might be provided from a SMDR message.

This complete set of data structures is replicated independently for each CTI server that monitors a separate PBX within the overall call center environment.

The call-centric structure and the list of participants facilitate a common framework for modeling the various types of complex call scenarios that may occur during the life of a call, far beyond the simplest example of a basic two-party telephone call. Moreover, the recording units can link references (i.e., logical pointers) to the audio recordings for a portion of the call, so that these audio sections are associated with the total history of the logical telephone call. Each call record can be linked within the database to an open-ended list of references, which provides: the name of a Voice Server; the name of a .WAV file containing the audio recording; the offset within the .WAV file to the start of the recording segment; the start time of the recording segment; and the duration of the recording segment.

Rather than relying exclusively upon the call identification number assigned by the PBX, the CTI server of the preferred embodiment obtains a Globally Unique Identifier (GUID), that is generated at the software's request by the underlying Microsoft Windows NT operating system, and uses that GUID to identify the call uniquely within the recording system's memory, online storage database, and offline storage archives. The GUID is initially requested at the start of the call. While the call remains active, the CTI server maintains a record of both the call identification number assigned by the PBX, and the GUID assigned to the call by the software of the preferred embodiment. When a CTI event arrives, the system searches the telephony model to find a matching call record for the PBX-assigned call identification number. At transition points during a call's lifetime, such as when it is transferred or conferenced, the PBX typically provides the old and new identification numbers together in that single transition event. In these cases, after locating the matching call record, the software of the preferred embodiment updates its record of the call identification number now being used by the PBX while retaining the originally allocated GUID value. In this way, the same GUID identifies the call throughout its lifetime, even while the PBX call identifier may be changing. The long-term uniqueness of the GUID value is also useful if the PBX recycles and reuses previously assigned call identifiers. It further helps in dealing with calls within a multiple PBX environment. While another PBX may coincidentally use the same call identification number, a different GUID is assigned at the start of each individual call, thereby avoiding a conflict within the telephony model.

24

As shown in FIG. 11, the CTI server consists of three distinct layers. Each layer actually runs in a separate thread of execution, and communicates with the other layers through shared memory, control semaphores, and message queues. The first layer 1110 is responsible for gathering input from the PBX data link(s), and there can actually be several threads running to provide better throughput capacity or to handle multiple diverse input sources (e.g., SMDR and real-time CTI messages). After saving the clock time when a message is received, the first layer 1110 places the message into a queue for subsequent processing by the second "analyzer" layer. The second layer 1120 is responsible for updating and maintaining the telephony model within the memory of the CTI server, and for deciding when to send copies of call records onward to the rest of the recording system. When a call record needs to be sent onward, the call record is placed into a message queue for subsequent processing by a third "message emitter" layer 1130, which is responsible for communications with other components of the overall recording system. This separation of layers gives the CTI server the flexibility to process its input and output sources in a de-coupled fashion, so that any delay in one area of communications does not affect the processing of another area. In a sense, the design approach provides a virtual "shock absorber" so that bursts of input traffic, or temporary lag times in communicating with other parts of the recording system, can be tolerated without loss of data or incorrect operation of the system.

The call records saved within the telephony model also include a record of the last state of the device as reported by the PBX. This information is used by the analyzer to run state machine rules, in order to select a handler routine for a subsequent message. The CTI server uses the previous state of the device (e.g., ringing, answered, and so forth) along with the current state of the device to select a handler routine from a matrix of potential choices.

The analyzer layer is of particular interest, since it is responsible for updating and maintaining the data model of telephony activity. Its overall program logic flow is illustrated in FIG. 12 and the subroutine called at step 1230 is shown in further detail by FIG. 13. This program logic is described below.

1. Receive a CTI event from the message queue at step 1228.
2. Enter the subroutine at step 1230 to update the telephony model. Referring now to FIG. 13, search the data of model of telephony activity, to find a matching record at step 1322 with the same monitored device (i.e., telephone set).
3. If the PBX-assigned call identification number does not agree, search for a matching record in the lists of calls on hold, in transition states, or recently completed. If a match is then found, move the call record on the affected device to the list of calls in transition states, and move the matching record to the monitored device.
4. At step 1324, use the previous state as recording within the telephony model, along with the new state reported in the CTI event, to select the appropriate handler routine at step 1332 from a matrix of choices. The handler routine will be one such as those described below.
5. At step 1340, run the steps of the handler routine. This will commonly include steps to save at step 1342 information from the CTI event into the call record, to update the call-related portion of the Object Status, if necessary (step 1344), to update Participants within the Object Status, if necessary (step 1352), to run additional action methods or handler routines for other affected telephony objects, if necessary (step 1348), and to post Object Status to the message Queue for the Emitter to a target platform (step 1354).

US 6,785,370 B2

25

6. At step 1360, returning to FIG. 12, at step 1232, discard completed calls within the data model of telephony activity, if they have aged beyond a certain re-configurable time limit.
7. Call the "hang-up" routine at step 1233 for any held call that have aged beyond a separate re-configurable time limit. Likewise, call the "hang-up" routine for any calls

26

marked in transition, which have aged beyond another separate re-configurable time limit.

8. Continue again from the beginning of this logical program flow at step 1226.

The following description lists processing steps for various handler routines that may be called in response to certain event types using a decision matrix based upon past and current state information.

Handler Routines	
Ignore:	adjust state based on CTI event
DialTone:	save the initial start-time of the call save the original dialed number, if available adjust state based on CTI event
RingIn:	adjust state based on CTI event event time-stamp when ring occurred clear call record set inbound, outbound, internal
Answer:	adjust state based on CTI event compute total ringing duration fill in call record with calling party & called party generate START message to recording system
Abort:	adjust state based on CTI event clear timers & original dialed number
Hang-Up:	adjust state based on CTI event update call record to stop all parties indicate which party actually hung up on the call generate STOP message to recording system
RingOut:	adjust state based on CTI event time-stamp when ring occurred (i.e., now) clear-call record set inbound, outbound, internal compute total ringing duration (i.e., zero) fill in call record with calling party & called party generate START message to recording system
Hold:	adjust state based on CTI event stop participant placing the call on hold add new placeholder participant for HOLD generate TRANSFER message to recording system move call record to hold area fill device slot with a new empty call record
Resume:	if device slot not idle, move call record to transition list move matching call record from hold area to device slot adjust state to "active" stop the placeholder participant for HOLD add new participant for telephone set that resumes the call generate TRANSFER message to recording system
Conference:	if call record found in hold area, if device slot not idle, move call record to transition list move matching call record from hold area to device slot adjust state to "active" stop the placeholder participant for HOLD add new participant for telephone set that resumes the call generate TRANSFER message to recording system adjust state based on CTI event add new participant for telephone set that is added via conference generate CONFERENCE-ADD message to recording system
Transfer:	if call record found in hold area, if device slot not idle, move call record to transition list move matching call record from hold area to device slot adjust state based on CTI event stop the participant leaving the scope of the call (either a device or HOLD) add new participant receiving the transferred call generate TRANSFER message to recording system
ConfDrop:	adjust state based on CTI event stop the participant leaving the scope of the call generate CONFERENCE-DROP message to recording system
OpAnswer:	adjust state based on CTI event re-compute total ringing duration correct the affected participant entry in the call record generate CORRECTED message
DestChanged:	clear call record the call will be processed via a subsequent CTI event

US 6,785,370 B2

27

The following step-by-step description describes the same call scenario as in FIG. 3, but with emphasis on the data model of telephony activity.

1. A real-time CTI message occurs describing that phone B is ringing, but not yet answered.
2. The "RingIn" routine is invoked.
3. The telephony model is updated with the time when ringing started (for use later in measuring ring duration) and the call direction. These facts are stored with device B 340.
4. A real-time CTI message occurs describing the start of the call between A 335 and B 340.
5. The "Answer" routine is invoked.
6. The telephony model is updated to reflect the initial 2 participants (A and B) started normally at t0 310.
7. A copy of the call record is sent onward to the rest of the recording system.
8. The call record is retained within the telephony model, associated with device B 340.
9. A real time CTI message occurs describing that B 340 placed the call on hold.
10. The "Hold" routine is invoked.
11. The telephony model is updated to reflect that B 340 transferred the call to HOLD 345 at t1 315. (This information is accumulated with the information previously gathered at t0).
12. A copy of the call record is sent onward to the rest of the recording system.
13. The call record is removed from device B 340 within the telephony model, but kept in a list of held calls.
14. A real-time CTI message occurs describing that B 350 returned to the call and invited C 355 by conferencing.
15. The "Conference" routine is invoked.
16. The call record is moved within the telephony model from the list of held calls back to device B 350.
17. The telephony model is updated to reflect that HOLD 345 transferred the call back to B 350 at t2 320. (Note that information is accumulated with the information previously gathered at t0 and t1).
18. A copy of the call record is sent onward to the rest of the recording system.
19. The telephony model is updated to reflect that C 355 joined the call as a conference participant at t2 320. (This information continues to be accumulated with previously gathered information).
20. A copy of the call record is sent onward to the rest of the recording system.
21. The call record is retained with both devices B 350 and C 355 within the telephony model.
22. A real-time CTI message occurs describing that C 355 dropped out of the call.
23. The "ConfDrop" routine 386 is invoked.
24. The telephony model is updated to reflect that C dropped out of the conference at t3. (This information continues to be accumulated with previously gathered information).
25. A copy of the call record is sent onward to the rest of the recording system.
26. The call record is removed from device C within the telephony model, but retained with device B.
27. A real-time CTI message occurs describing that A terminated the call.
28. The "Hang-Up" routine is invoked.
29. The telephony model is updated to reflect that A stopped normally and B stopped because the other party hung up at t4 330. (This information continues to be accumulated with previously gathered information).
30. A copy of the call record is sent onward to the rest of the recording system.

28

31. The call record is removed from device B 350, but kept in a list of completed calls.
32. A SMDR message occurs summarizing the call in its entirety.
33. The list of completed calls is searched to find a match, and the appropriate call record is retrieved.
34. The call record is updated with the trunk channel information from the SMDR message.
35. A copy of the call record is sent onward to the rest of the recording system.
36. The call record is removed from the list of completed calls.

FIG. 14 depicts the flow of information within the remainder of the recording system. The same enhanced search information S1 1412 is provided by the CTI server to all of the recording units involved in handling a portion of the call. Even if a call is transferred to another telephone set, which is attached to an input channel on a different recorder, the entire call will still remain associated as one entity within the system. Each recorder maintains a local copy of the audio sections V1 1416, V2 1420, and V3 1424 that it obtained during the call, along with a complete call record containing search information S1 1412 which contains the two-tiered call and participant model. The search information is copied to a central database server 1450, along with references (i.e., logical pointers) to the original audio recordings VR1 1428, VR2 1432, and VR3 1436. When a user searches for a call, the search results 1465 will include the complete call record S1 1412. By using the audio references the playback software can reassemble the complete audio for the original call, including sections possibly obtained from different physical recording units.

The general principles behind the method described above would be suitable, not only for representing the complete history of telephone call's lifetime, but other forms of multi-party communications. This may include certain forms of radio traffic that have an associated data link, which provides "talk group" identification numbers (or similar types of descriptive search data in relation to the audio traffic).

Call Record Generator

The Call Record Generator (CRG) in accordance with the present invention performs the function of combining voice and data into call records. It performs this function at or near real time. The CRG, when combined with the metadata normalization module CTI Server, makes up a system that can be used in current and future communication recording products.

The CRG is responsible for collecting data from different sources with respect to portions of a call on various recording input channels, and merging them together into a unified call record. One of these sources is the recorder that creates the files containing media. Another source provides metadata describing the when, who, why and where information of a call. This call record metadata comprises the start and stop times of a segment within a call, as well as CTI data such as telephone numbers and agent IDs. These metadata sources include but are not limited to Telephony switches and Trunked Radio servers. The CRG depends upon the CTI Server to normalize data from these sources.

FIG. 1 illustrates the relationship between the CRG and the rest of the system. Since call records are an essential part of the recording system, there is one CRG dedicated to each recorder and physically located in the same Voice Server. If other system components become inoperable, call record generation will remain functional (albeit at a reduced level).

US 6,785,370 B2

29

The CTI server supplies switch events to the appropriate recorder indicating either the status of calls or providing data for population. The CTI server provides, along with call record data, the association between the recorder location (i.e., Voice Server and recording input channel number) and the switch connection point. The switch connection point is described as either the extension for extension side recording or the Trunk ID/virtual channel (TDM time slot) for trunk side recording. In addition to this mapping, an agent identification will be supplied for agents currently associated with this call. The recorder location, switch identification and corresponding agent are stored in the call record. The CRG is designed to work with many different configurations of the disclosed system. These configurations include: systems without CTI Servers; systems with Real-time CTI Servers; systems with non-Real-time CTI Servers; recorders with analog inputs; recorders with digital inputs; recording on the trunk side of the telephony switch; and any combination of CTI Servers, Recorder inputs, and recorder positions mentioned above.

Due to the non-standard operation of telephony switches and flexibility requirements of the recording device, the CRG must handle event data arriving in different chronological order. In accordance with a preferred embodiment, it accomplishes this by requiring all events to indicate time of occurrence and maintaining a history of them. A call record can be created solely from either event sources but when both are present, call records are generated using recorder information together with CTI data.

It is clear that the use of different data sources and non-synchronous messages, as required to support various alternative configurations of the overall system, add considerable complexity to the CRG. For example, with the many different objects supplying information for a particular call, the messages from each can be received in any order. The CRG must be able to accommodate this requirement. In some configurations, objects supply redundant information to the CRG. The CRG provides a mechanism for selecting which information will populate the call record.

In the most basic mode of operation, the CRG has no CTI input and is recording solely on VOX events from the recorder controller (the term "recorder controller" is used interchangeably herein with "Audio Recorder"; both terms refer to the software that primarily directs the processing of the audio data). VOX is Dialogic Corporation's digital encoding format for audio samples. This term is also sometimes used to refer voice-activated initiation of recording, a process that conserves storage space since a continuous recording process would include periods of silence. These VOX events mark the beginning of energy activity on a phone line and are terminated by the lack of activity. With this approach, an actual phone call may include several call records. To address this problem, the recorder waits a configurable holdover period while silence is present before terminating an active VOX clip (the term "Recorder" is used interchangeably herein with the term "Voice Server"; both terms refer to the physical recording server). The goal is to concatenate parts of a phone call where gaps of silence exist. The solution lies in determining an appropriate holdover time so as to avoid merging audio from the next phone call if it occurs close to the end of the last call.

The next level of operation is where the recorder hardware can detect telephony signaling such as off hook and on hook. The CRG has no CTI input from the switch and is recording solely on events from the recorder controller, but these events mark the beginning and end of a phone call (off hook and on hook). The resultant call record reflects a phone

30

call in entirety but lacks much descriptive data that accompanies switch data.

The highest level of operation involves the use of a CTI Server. In this configuration, the CRG receives recorder events as well as CTI events. Since CTI events give the CRG a description of the entire phone call, information obtained from them drive the creation of call records. Recorder data describing audio events are absorbed into the CTI call record whenever audio and CTI times overlap. With CTI events driving call record generation, non-audio based call records can be created.

Mixing of recorder and CTI data occurs by comparing ranges of time indicated. For example, a person whose telephone extension is being recorded is involved in a phone call for a given period of time. The recorder events indicating that audio was recording on the same extension during the same time period are associated with the CTI metadata for that phone call. Since the data from the CTI Server may arrive before or after the corresponding recorder events, the CRG maintains an independent history for each type of data.

For the case where CTI events arrive before the recorder events, the CTI events are added to the CTI history list. When the corresponding recorder events arrive, the CTI history list is swept for matching time ranges and associations are made when they occur. For the case where recorder events arrive before the CTI events, the recorder events are added to the recorder history list. When the corresponding CTI events arrive, the recorder history list is swept for matching time ranges and associations are made when they occur.

Previous recording systems stored voice data and metadata in separate locations. A significant disadvantage to this approach is that it is left up to the other software subsystems to combine the information when required. This approach makes the work of other system features, such as playback and archiving to offline storage, more complicated and prone to error. By performing this "early binding" of the audio and CTI data in accordance with the present invention, such problems are avoided and the above desirable features are therefore much simpler to implement in a correct, robust fashion.

When attempting to playback media for a given call record, the playback mechanism must figure out where the audio for the call record exists and when determined, retrieve and locate the start time inside this media. The CRG places this media metadata in related tables, thus informing the playback mechanism what files are associated, their location, and what time ranges inside the file are available for playback.

Most communication systems require an archive mechanism to store large amounts of data that cannot be kept online due to capacity limitations. The CRG used in accordance with this invention assists with archiving by allowing both call record metadata and the media files to be stored on the same offline media. Current versions of recording systems store call record metadata and media files on separate offline media making restore operations more complicated.

For enhanced security purposes in a preferred embodiment, the CRG accesses media files associated with a call record through the use of media segmentation. A media segment includes, in addition to a media filename and location, a start time and duration inside the media file. Media segmentation is necessary when creating CTI based call records since a call record may involve many recording locations throughout the life of the call. The specified time range isolates a portion of the media file that can be accessed

US 6,785,370 B2

31

through this call record. This feature is very important when there are many call records located in one media file. A user attempting to play back media of a call record, to which he has the permission for access, may or may not have permission to play back other call records sharing the same physical file.

The Call Record Generator is responsible for merging CTI search data and a multitude of voice recording segments together into a single manageable unit of data. This software includes a flexible receiver algorithm to allow voice and search data to arrive in either order, without requiring one to precede the other. Once combined, the call record can be managed as a single entity, which greatly simplifies and reduces the work necessary to perform search, retrieval, and archival operations. This approach also offers a more natural and flexible framework for controlling security access to the recordings, on an individual call basis (or even on selected portions within a call).

As shown in FIG. 15, a recording unit operating with only voice signaling to guide the creation of its call records could make a number of fragmented audio segments. When the recording unit is supplied with CTI search data giving a complete history of the call's lifetime, and when it is designed to merge the CTI search data and audio segments into a combined unit of Voicedata™, the results can simplify and reduce the work necessary for a user to obtain a desired call from the system. Several audio segments can be grouped together, and can be understood by the system as being part of the same logical telephone call. It is also possible that a single audio segment was recorded, even though parts belong to separate telephone calls, because the delay between stopping the first call and starting the second call was very brief. Without a sufficient silence gap, it may appear to the voice recording unit that this was a continuous segment of audio, rather than belonging to two separate calls. When the CTI search data is merged with the audio segments, the system can use this information to recognize when an audio segment should be split and divided between two logically distinct calls.

The purpose of the Call Record Generator (CRG) is to collect information describing multimedia data and store it in a central location. The CRG produces Master Call Records (MCRs) that encapsulate information describing a phone call as well as the location multimedia that is associated with it. This description data comes from a multitude of sources including but not limited to a Voice Server and CTI Server. Likewise, the design of the system envisions that there will be a number of possible input sources for audio recording.

Whatever the means for collecting CTI information, it is communicated to the rest of the system in a common, normalized format. The CTI information is passed from the translation modules to a message router. From that point, copies of the information are sent to the scheduling and control services and to the CRG for the appropriate recorder (s). The scheduling and control services are responsible for starting and stopping the audio recorder, according to pre-defined rules that are dependent upon time and CTI information. The CRG is responsible for merging the audio recording with the CTI information to determine the temporal boundaries of the call and prepare the Voicedata for storage.

The user workstation typically searches and retrieves records from the Voicedata storage, and then obtains audio for playback directly from each recorder's private storage area. The user workstation can also be used to monitor "live"

32

conversations by communicating directly with the recorder. The user workstation can also control the audio recorder indirectly by manipulating the rules used by the scheduling and control services.

In the preferred embodiment, the user workstation has software that is configured to display a graphical user interface (GUI) such as that shown in FIG. 16. The GUI in FIG. 16 uses the information compiled in the Master Call Record to generate a graphical representation 1610 of the call, as well as displaying the call record information in alphanumeric form in a table 1620. Further, when the call is played back, the displayed segments in the graphical representation are highlighted to indicate the portion of the call being played back. For example, in FIG. 16, if the entire call is played back, when the portion of the call that occurred between 6:20:08 AM and 6:55:31 AM is played back the bars 1632, 1634, and 1636 are highlighted from left to right as the call is played back. Thus, as the part of the call that occurred at 6:55:31 AM is reached, bar 1634 is fully highlighted, and bars 1632 and 1636 are highlighted starting from the left and extending to those points on bars 1632 and 1636 that are directly above and below the right-hand endpoint of bar 1634. After the played back call reaches the part that occurred at 6:55:31 AM, the bar 1638 begins to be highlighted starting at the left endpoint. When the part of the call that occurred at 7:10:22 AM is reached, the bar 1636 is fully highlighted. At that point, the bars 1632 and 1636 are highlighted from their left-hand endpoints and extending to points directly above the right-hand endpoint of bar 1638. The process continues as long as the call is being played back, until bars 1632, 1634, 1636, 1638, 1642, and 1644 are completely highlighted.

In alternate embodiments of the subject invention, playback of a portion of a call can be activated directly from the graphical view by mouse-clicking or by selecting from a pop-up menu; circular "pie-charts" show the percentage of time for each party involved during the lifetime of the call; an animated vertical line scrolls along to indicate the progression of time when the call whose graph is being displayed is played back; and miniature pictorial icons are shown within the graphs to indicate start/stop reasons, type of participant, etc. All of these embodiments are enabled by the data contained in the Master Call Record.

As a method of managing complexity, the preferred embodiment of the system uses data abstraction to isolate the internal details of certain structures to those components which need to operate directly upon them. Information is organized by the collectors (or producers) of that data, into a digested form that is more easily usable by the applications which need to retrieve and process the data.

For example, the CTI translation modules supply normalized records to the rest of the system in a common shared format, rather than exposing the details of various different CTI links. The system data model is call-centric, containing a detailed cumulative ("cradle to grave") history, rather than event-centric, which would place the burden of work on the receiving applications. Likewise, agent information is session-oriented rather than event-oriented.

Whether collecting information from a CTI link, or recording audio from a telephone call, a fundamental design advantage for the system of the preferred embodiment that it operates virtually invisibly, from the end-user's perspective. The system architecture is designed to avoid any interference with the normal operation of a call center environment.

For example, the CTI translation modules are focused exclusively on collecting and normalizing information that

US 6,785,370 B2

33

is to be supplied to the rest of the system. Liability recording systems, and quality monitoring systems that use "service observance" techniques, do not require any active call control on the CTI links. Only the technique known as "dynamic channel allocation" requires active call control through CTI links to establish a "conference" or "bridge" session between the audio recorder and the telephone call participants. When active control is required to implement such a feature, it can be implemented through a new logically separate task, without significantly affecting the rest of the system design. For customers that have existing CTI infrastructure and applications, the system will not interfere with their existing operations.

The CRG is responsible for collecting data from the CTI Server, creating CTI-based call records, and attempting to match those records with existing recorded audio data. If the CRG receives CTI information indicating that audio data for the same call resides on two or more recorders (for example, due to a transfer), records will be generated for each portion with a common call record ID. This ID can later be used to query for all of the pieces ("segments") comprising the complete call. Each segment will identify the recorder that contains that piece of the call.

During playback, a player module connects to a program located on a Voice Server called the Playback server ("PBServer"). The machine name of the particular Voice Server which holds an audio segment is stored by the CRG in the call record table within the Voicedata storage, and is passed into the player module after being extracted by a User Workstation's sub-component known as the call record browser. A call record playback request is then submitted, which causes the PBServer to query for a specific call record's audio files located on that physical machine, open them, and prepare to stream the audio upon buffer requests back to the client software (the player module) on the User Workstation. If successful, a series of requests is then issued from the client, each of which will obtain just enough audio to play to a waveOut device while maintaining a safety net of extra audio in case of network delays. Upon a request to "move" within the scope of a call record, the PBServer repositions its lead pointer to the desired location and then begins passing buffers from that point. This series of Request and Move commands continues until the user chooses to end the session by shutting down the client-side audio player.

As used herein, the term "Call Control" refers to the part of the metadata concerning the creation and termination of call records. The term "Media" refers to the actual data that is being recorded. This term is used interchangeably with audio since the primary design of the CRG is to support audio recording. However, the CRG could apply to any data being recorded including multimedia or screen image data. The term "Metadata" refers to informational data associated with multimedia data describing its contents. The term "Call Participant" refers to an entity that is involved in a phone call. There are at least two participants involved in a call; namely the calling and called parties. Participants can consist of people, VRUs, or placeholders for parties being placed on hold. The term "Recorder Participant" refers to a participant in the MCRs Participant list who is located at the same connection point on the Switch to which the recorder input channel is connected. In accordance with the present invention, there can be more than one Recorder Participant associated with a call record since participants can enter and leave many times in a call. For any given recorder channel, there can only be one matching Recorder Participant active (not disconnected) at any given time across all call records

34

associated with that channel. A "VOX-based Master" Call Record contains information contributed by events from the Recorder alone, in the absence of data from a CTI Server. A VRU is a Voice Response Unit: an automated system that prompts calling parties for information and forwards them to the appropriate handler.

Once a recorder channel becomes involved in a phone call, it will be associated with all subsequent CTI events pertaining to the same call. This occurs even if the recorder location is no longer involved in the call. As an example, consider a phone call involving a transfer. FIG. 16A shows the subject system containing a CTI Server 710 and Recorder 1640. A recorder channel 0 1650 is attached to the extension side to extension 0001 1622. A phone call is initiated from the outside by some agent "A" 1602 and initially connects to agent "B" 1608 at extension 0001 1622. Agent "B" 1608 places "A" 1602 on hold and transfers him to Agent "C" 1612 at extension 0002 1630. The CRG recording extension 0001 1622 would receive all update messages with regard to this call since he/she participated in the call. Descriptive information from the CTI Server 710 would look like that in table 1600 in FIG. 16B. Audio clips recorded while agent "B" 1608 was involved in the call are recorded in a VOX based call record as shown in FIG. 17. The three media files created from the conversation may overlap with the recorder participant (agent "B"). At some point, determined by the order by which recorder and CTI events are received, audio data information from the VOX call record is absorbed into the CTI MCR for the times the recorder participant is involved (see the results after the sweep of the VOX and CTI history lists). For this call record, audio recorded between times t_1 and t_4 is absorbed. Any remaining audio is left in the VOX MCR for possible absorption in other CTI MCRs adjacent in time to this one. Since extension 0001 in this call record is different from the other participants in that it is associated with the same switch point as the recorder channel, he/she is referred to as the Recorder Participant. From time t_4 and on when the Recorder Participant is no longer involved in the call, CTI events are still received for that channel. This allows the system to supply information about the entire phone call involving extension 0001 that may be of interest to the customer.

Since the CRG must be prepared to handle messages from different components arriving in any order, it is designed to collect information in separate structures. Depending upon the operating mode of the CRG channel, call records are created from information collected in one or more of these repositories. The name given for these structures is Master Call Record (MCR).

The major components of the preferred embodiment contributing information for call records are the Recorder and the CTI Server. In alternate embodiments of the subject invention, other multimedia or screen image data may be provided to the CRG in order to be merged with descriptive metadata.

Recorder events are assembled into VOX MCRs identified by a unique sequence number. Individual events contain a sequence number identifying a specific structure to update (or create). For example, a recorder event would be used to indicate the beginning of a new audio segment. While that segment is active, other messages containing the same sequence number are used to add metadata to the audio segment. These update events include, without limitation: DTMF digit data; agent association information; change of audio filenames holding future audio data; selective record control; and ANI, ALI, DNIS information. DTMF is Dual Tone Multi-Frequency and refers to sounds emitted when a

US 6,785,370 B2

35

key is pressed on a telephone's touch-tone keypad; ALI is Automatic Location Identification, a signaling method that identifies the physical street address of the calling party and typically used to support Emergency 911 response. Finally, a disconnect message identifies the end of an audio segment.

Events received from the CTI Server are accumulated in CTI MCRs. Each event received from the CTI server contains a unique identifier. Events containing the same unique identifier are associated with the same CTI MCR. If any VOX MCR contains audio data that overlaps in time with Recorder Participants in a CTI MCR, then that audio data is transferred to the CTI MCR. If the absorption process causes all audio metadata for a VOX MCR to be consumed, the VOX MCR is deleted from the VOX list. Therefore, call records generated on the same channel will never have overlapping audio data. VOX MCRs containing leftover audio not absorbed by CTI MCRs are either be saved into the central database if of significant duration or discarded.

Data from a Master Call Record alone is processed into call record(s) that populate the system's central database. Thus, if the recorder channel is set up for VOX based recording only or if the CTI Server is down, VOX MCRs drive call record creation in the system. Otherwise, the CTI MCRs drive call record creation in the system.

The VOX and CTI MCR structures are maintained in two separate lists for each recording input channel. These are the VOX History List and CTI History List respectively. These lists represent a history of call activity sorted chronologically. The depth of the history list is driven by a configurable time parameter indicating the amount of history that should be maintained. By maintaining a history, the CRG tolerates events received in any order as long as received within the time boundaries of the history list. Some CTI Servers obtain data from SMDR type switches which report entire phone calls at the end of the call with a summary message. Maintaining a history buffer for VOX MCRs allows us to hold onto audio data for a period of time to allow later CTI summary messages to consume (absorb) the associated audio.

The MCR has status fields associated with them indicating its current state. At an installation involving real time CTI events, when a recording input channel receives a CTI event, it may indicate that a participant connected at the same telephony switch location as the recorder (Recorder Participant) is active in the call. The MCR is considered active as long as there is a Recorder Participant still active in the call. During this period, any new audio arriving on this channel is associated with the MCR. When a Recorder Participant leaves the call, the MCR becomes inactive. Since any Recorder Participant can become involved in the conversation at any given time through transfers or conferences, the MCR can transition into and out of active state many times throughout the phone call.

Another field in the MCR indicates the overall status of the call. This flag, called `m_bComplete`, indicates when the phone call is over. An MCR is considered incomplete as long as there is at least one participant still active in the call. When there are no participants active in a MCR it is considered to be complete. Therefore, calls created in real-time will start as incomplete and at some point transition into completed state. When an MCR enters complete state, a Closed Time variable is set to the current time. This time is used in maintenance of the History List. A closed MCR is allowed to stay in the History list for a configurable amount of time before it is deleted. During this window of time, events arriving out of timely order are allowed to update the

36

MCR. Once this configurable amount of time expires, the MCR is updated in the local database, marked complete, and deleted from the History List.

When the CRG starts, it initializes, for each recording input channel, a location which identifies where it is attached to the telephony switch. Each recorder location contains status fields describing the state of the switch and CTI server involved. These fields are `m_SwitchStatus` and `m_MetadataserverStatus` respectively and are set to "down" state until an event is received that indicates otherwise. When a message is received indicating a change of state, all associated recorder locations are updated with the new state value. Any changes in operation are processed upon receipt of the next event for the channel.

Another configuration setting indicates what type of external sources are allowed to populate call records created on a record channel. This setting, `m_Extremetadatasource`, is set to zero when a record channel is to be driven by recorder events only. It is set to non-zero when external events are allowed to generate MCRs.

The CRG is able to react to a variety of situations that may arise. For example, when the CRG first initializes and a record channel is configured to receive CTI input, how are call records generated if the CTI server is not running? What if the CTI Server is running but the communication path to the recorder is down? The CRG must also be able to react to external parts of the system, that it normally relies on for input, being temporarily unavailable for periods of time. In accordance with a preferred embodiment, the CRG handles these situations by operating in different modes: Initial, Degraded, and Normal. These modes are applied individually to each channel in the recorder.

Initial Mode: When a recorder starts up, there can be a considerable amount of time before the rest of the system becomes operational. The CRG must be ready to handle events coming from the Recorder immediately after startup. Therefore, the CRG must be ready to accept recorder metadata without supportive information from the CTI server. VOX MCRs are created from these recorder events and are stored in the VOX History List. When VOX MCRs are completed, they are made persistent in the Local Data Store.

The CRG system will remain in this mode until all of the following conditions occur: (1) the CTI server becomes available; (2) the switch being recorded by this channel becomes available; and (3) a configuration option for the channel indicates it is to be driven from an online CTI server and switch.

Degraded Mode: If a record channel is configured to be driven from a CTI source, only CTI MCRs are entered into the database. These CTI MCRs absorb any recorder metadata that intersects with the time ranges of the CTI events. No VOX MCRs are made persistent. If, however, the CRG detects that the CTI Server, switch, or associated communication paths are down, the channel enters Degraded mode. This mode is similar to Initial mode in that VOX MCRs are made persistent when completed. Any CTI MCRs that were left open at the time the CTI Server went down are closed and updated for the last time. The recorder channel will remain in this state until the three conditions indicated in "Initial Mode" are met. Only then will the recorder channel transition into Normal mode.

Normal Mode: Under normal operating procedures in a system with a CTI server and switch online, MCRs are created whenever a VOX or CTI connect event is received

US 6,785,370 B2

37

and stored in the appropriate list. For each VOX message received, the CTI History List is swept to see if audio metadata can be absorbed by a matching MCR. Any remaining audio data is placed in a VOX MCR. For CTI events involving updates to Recorder Participants, the list of VOX MCRs is swept to see if audio metadata can be absorbed. CTI MCRs are made persistent to the Local Datastore when first created, upon significant update events, and when completed. VOX MCRs are not made persistent to the Local Datastore as they should be completely absorbed by CTI MCRs. There is a configuration parameter that can enable leftover VOX MCRs to be made persistent when they are removed from the VOX MCR history list.

Transitions from Initial/Degraded to Normal Mode: When a CRG channel is in Initial or Degraded mode, VOX MCRs are recorded into the Local Data Store when completed. If notification is received indicating a recorder channel meets the three criteria indicated in "Initial Mode", the channel is set to Normal mode. From this point on, only CTI based MCRs are made persistent and VOX MCRs will be absorbed by the VOX events. Since CTI events represent an accumulated history of a phone call, prior events occurring while the connection between the CRG and CTI Server was lost (or was not yet established) are nonetheless summarized in each update message. The time spans of Recorder Participant(s) are compared to audio data in the VOX MCR list, with any overlaps causing the audio data to be absorbed. In this way, any audio data that occurred while a connection to an external component is temporarily unavailable will still be capable of being correctly associated.

Transitions from Normal to Initial/Degraded Mode: When the CTI server and switch becomes available for driving the call record creation and processing, the CRG channel enters into Normal mode. A heartbeat message is used to periodically update the status of the switch and CTI Server. When the heartbeat is lost or there is a message indicating one of the components has gone down, the recorder channel switches to Degraded mode. The CRG will still create and maintain MCRs in the VOX list and force MCR closure on open CTI MCRs as they pass out of the CTI history buffer. The sweeping action of audio metadata among incomplete CTI MCRs will cease, preventing all future audio data from being absorbed by it. VOX MCRs are made persistent in the database when they leave the history buffer.

Trunked Radio Mode: In an alternate embodiment of the subject invention, fields in the call record structure are added to support trunking radio. Information contributing to these fields may be obtained from communications with a Motorola SmartZone system. This system uses the Air Traffic Information Access (ATIA) protocol to communicate metadata related to radio activity. The embodiment has a trunking radio server similar to the CTI server that provides an interface between the SmartZone system and the recorders of the preferred system. This server provides the normalization of data and distribution to the correct recorder. There are currently two modes of operation of the Motorola trunking radio system that are discussed below.

Message Trunking: In this mode, when a radio is keyed, it is assigned a particular frequency to communicate on. When the radio is de-keyed, a message timeout timer (2-6 seconds) is started. If another radio in the talk group keys up during this time, the controller uses the same frequency for transmission and resets the timer. The conversation will remain on this frequency until the timer is allowed to expire. During this time, all events that are reported with respect to this conversation will have the same call number associated with them. Therefore, the concept of CTI based call records with many participants has been applied to Message Trunking.

38

If the timer is allowed to expire, future radio transmissions will be assigned to another frequency and call number. The server needs to detect this occurrence and properly terminate a call record.

Transmission Trunking: Transmission Trunking does not use the holdover timer mechanism used in Message Trunking. When a radio is keyed, it is assigned a particular frequency for transmission. When de-keyed, the channel frequency is immediately freed up for use by another talk group. Therefore, a conversation can take place over many channels without a call number to associate them. The concept of VOX based call records which contain one radio clip per MCR is used in this mode.

Selective Record: There may be certain phone calls involving extension or agents that are not to be recorded. Selective Record is a feature that tells the system to refrain from recording a call while a certain condition exists.

Virtual CRG: MCRs can exist in the subject system's database that have no audio associated with them. These non-audio MCRs can be created due to different features of the subject system. Some customers may require that all CTI data coming from their switch be saved even though they are not recording all extensions or trunk lines. By creating records from the CTI data alone, in the absence of recorded audio, this mode of operation can provide the customer with useful information for statistical analysis or charting purposes. Likewise, records created based upon CTI data alone may provide a useful audit trail to verify the occurrence of certain telephone calls, analyze traffic patterns, or to perform other types of "data mining" operations. In that case, a CRG is associated with the CTI Server mechanism to receive all CTI events that are not matched to a specific recorder. These CTI MCRs are made persistent to the Central Database upon call completion.

Call Record Structure: Call record start and stop events originate from two independent sources: the Recorder and the CTI server. The CRG must perform some method of merging events from these two sources in such a way that the resultant call record contains the best information available. CTI server events are advantageous in that they provide more information than the recorder and can also accurately determine a call record boundary. Recorder based events are a subset of CTI server events and can only distinguish call record boundaries based upon VOX or off/on hook. The recorder has advantages in that since it is in the same box as the CRG, receipt of these events is guaranteed as long as the recorder is running. The main purpose of the assembly process is to leverage the information coming from the CTI server in such a way that the entire phone call is assembled into one Master Call Record (MCR). The structuring of call records is weighed towards trunk side recording with the services of the CTI server driving call record creation. This type of configuration enables the system to summarize phone calls in the most effective manner. The manner in which the structure of the MCR designed to achieve this goal is discussed below.

Master Call Record: The MCR holds information accumulated for all events received necessary for archiving to the local data store. It consists of individual fields that are global to the entire call record as well as lists of specific information. Global information includes identifiers for the call record, the start and stop times of the entire call, the recorder location with respect to the switch, and flags indicating the call record status.

Lists included with each MCR contain the following information: Media File List—List of media filenames that

US 6,785,370 B2

39

make up the call (e.g., telephone or radio communications); Screen Data Capture File List—List of screen image files associated with audio on this channel; and Participant List—List of participants involved in this call.

40

The MCR is populated from events received from the CTI Server and Recorders. The following table shows the fields in the MCR, in a preferred embodiment, their data types, description and if they are stored in the database.

Master Call Record structure.			
Name	Type(max length)	Archive	Description
m__CallRecID	string†	Y	Unique ID (UUID) pertaining to entire call (Cil and Trunk Radio server provides same ID for call parts that are related to the same conversation.)
m__MetaDataSource	BYTE	Y	Indicates the source used to populate call record information. 0 = none 1 = CTI 2 = Trunking Radio
m__bCallComplete	bool	N	Indicates the end of a call. (i.e., there are no more active participants involved)
m__bCall-HoldoverExceeded	bool	N	If true, MCR has been in a complete state for a time period exceeding the configured Call Holdover time.
m__bMetadataHoldover__Exceeded	bool	N	If true, MCR has been inactive for a time period exceeding the configured Metadata holdover period. Used to allow completion of MCRs that haven't been updated for long periods of time possibly because of missed events.
m__bLastUpdate	bool	N	true when the CRG has decided to send the last update of this MCR. Used to prevent any future updates.
m__bDontArchive	bool	N	Indicates whether this call record is to be archived by data store. Certain record features such as selective record may prevent us from storing this call record.
m__CallDirection	BYTE	Y	Indicates call origin Outbound = 0x12, Inbound = 0x21, Internal = 0x11, Unknown = 0x44
m__CustomerNumber	string†	Y	Variable length character array dedicated to information the switch may provide with the call. For custom call record support. (e.g., account number)
m__pRecLoc	RecorderLocation	N	Pointer to recorder location descriptor associated with this channel. (see RecorderLocation class)
m__SSFile	list†	Y	List of TimestampedFilename (see below) objects representing Screen Data Capture filename(s) associated with a call record
m__Participants	list†	Y	Array of CallParticipants (see below) describing all participants involved in the call
m__XactionSema	HANDLE	N	Semaphore used to lock this MCR from being modified by any other threads.
m__SemaTimeoutVal	unsigned long	N	Maximum time thread is blocked on m__XactionSema access before returning.
m__bModified	bool	N	Set whenever MCR is changed in a way that requires update to the Local Data Store.
<u>VOX Call Record (Derived)</u>			
m__dwVoxCrNum	DWORD	N	Sequence number of first VOX MCR associated with this CTI MCR (if applicable).
m__bVoxInProgress	bool	N	Indicates this VOX clip is still active (i.e., End time is default time.)
m__CreationTime	time_and_date†	N	Holds time at which the call record was created. Used for debugging purposes to measure how long a call record is alive.

US 6,785,370 B2

41

42

-continued

Master Call Record structure.			
m_CloseTime	time_and_date †	N	Local time at which MCR was marked complete. Used for determining when call record is ready for archive.
m_MediaFiles	list	N	List of TimestampedFilename classes representing multimedia files used to store data with respect to this call record.
m_CtiInfo	CtiInfo	N	Class containing CTI type data associated with call record.
<u>Base Call Record (Derived)</u>			
m_wVersion	WORD	N	Version number of call record.
m_StartTime	time_and_date †	Y	Start time of call record
m_EndTime	time_and_date †	Y	End time of call record
<u>RecorderLocation</u>			
m_MetadataServerStatus	BYTE		Indicates the status of the metadata server driving call records for this particular recorder location. This source is in most cases the CTI server but can be other servers such as Trunking Radio Server 0 = "down", 1 = "up"
m_SwitchStatus	BYTE		Indicates the status of the telephone switch providing call record information for this particular Recorder Location. 0 = "down", 1 = "up"
m_ExtarnMetaDataSource	BYTE		Indicates what external source (if any is contributing call record meta data for this channel 0 = None (recorder only) 2 = CTI server
m_ChanID	ChannelIdentifier		Class identifying recorder channel.
m_SwitchID	Switch Identifier		Class identifying switch connection point.
m_SwitchChars	SwitchCharacteristics		Class identifying characteristics of switch needed by CRG.
<u>SwitchIdentifier</u>			
m_SwitchNum	WORD		Number identifying switch
m_wTrunkID	WORD		Identification of trunk line attached to switch. (Valid only if not equal to -1)
m_dwVirtualChannel	DWORD		Identifies time slot of digital line (T1 or E1) of interest. (Valid only if TrunkID is not equal to -1)
m_Extension	string† (6)		Extension number (Valid only if m_wTrunkID equals -1)
<u>ChannelIdentifier</u>			
m_wNode	WORD		Unique number used to distinguish between multiple Voice Servers.
m_wChannel	WORD		Unique number used to distinguish between multiple recording input channels with in a Voice Server.
m_bSignalSupport	bool		Indicates if hardware associated with this channel supports on/off hook signaling.
<u>SwitchCharacteristics</u>			
m_bTimeSynced	bool		Indicates if switch is synchronized with the system.
m_bRealTime	bool		Indicates if switch provides CTI info in realtime (true) or

US 6,785,370 B2

43

44

-continued

Master Call Record structure.		
m_iCmdTimeOffset	int	batched and sent periodically (false) Value that indicates any known time offset between events received at the switch versus the time the similar signal is received at the recorder. This value will be used to adjust CII generated timestamps before comparing to recorder events
m_iSwitchTimeOffset	int	For switches that are not time sync'd with the system, this value indicates any known time offset between the switch and the system time. This can be utilized if has some way of updating the time delta between switches and our system on a periodic basis.
<u>CtiInfo</u>		
RingLength	WORD	Time (in sec) between first ring signal and off hook.
DTMFCode	string† (50)	DTMF codes entered during conversation

TimeStampedFilename

Name	Type	Description
m_AFStartTime	Time_and_date†	Start time of audio file
m_StartTime	Time_and_date†	Start time of interest
m_EndTime	Time_and_date†	End time of interest
m_SegStartTime	Time_and_date†	Start time of segment inside file absorbed by this MCR.
m_SegEndTime	Time_and_date†	End time of segment inside file absorbed by this MCR.
m_PathName	string† (36)	Path describing the Voice Server and directory location where the audio files are located.
m_File Name	string† (36)	GUID-based name that uniquely identifies a specific audio segment's recording file.
m_wFileType	WORD	bitmap indicating types of media associated with MCR. bit Data 0- Audio Present 2- FAX Present 3- Video Present 3- Screen Capture data Present
m_wFileFormat	WORD	Recording format of media data, as defined by Microsoft Corporation's multimedia description file "mmreg.h"
m_bNew	bool	Used by local data store to indicate whether this record should be inserted (true) or updated (false) into the database.
m_bDiscard	bool	If true, don't allow playback or archiving of this media. Used for the Selective Record feature.
m_dwVoxCrNum	DWORD	Sequence number corresponding to VOX call record that provided this media.
m_iAssocPart	int	Index of Recorder Participant in the Participant list causing this media file to be associated with this MCR.

-continued

Master Call Record structure.		
<u>CallParticipant</u>		
m__AgentID	string† (24)	Registered ID of agent at extension (CTI) or Radio Alias (Trunking Radio).
m__Number	string† (24)	Full telephone number of the participant (i.e., ANI, DNIS)
m__Console	string† (10)	Seating position of participant that can consist of one or more stations (CTI) or Talkgroup ID (Trunking Radio).
m__Station	string† (10)	Unique telephone set. Possibly with multiple extensions
m__LocRef	BYTE	Describes the location of participant with respect to the switch. (1 = internal, 2 = external, 3 = unknown)
m__SwitchLoc	SwitchIdentifier	Class identifying the position of a participant relative to the telephone switch.
m__StartTime	Time_and_date†	Time participant joined the call
m__EndTime	Time_and_date†	Time participant left the call
m__ConnectReason	BYTE	How participant joined the call NotConnected = 0, NormalStart = 1, ConferenceAdd = 2, TransferRecv = 3, UnknownConnect = 9
m__DisconnectReason	BYTE	How participant left the call NotDisconnected = 0, NormalEnd = 1, ConferenceDrop = 2, TransferAway = 3, OtherPartyHangup = 4, UnknownConnect = 9
Changed	Bool	Indicates if recent change in CTI message. (not archived)
<u>Trunking Radio Only Information</u>		
SourceSiteID	BYTE	Site number that is currently sourcing audio on active call.
ZoneID	BYTE	Zone at which participant is currently located.
CIUNumber	BYTE	Console Interface Unit. Translates 12kbit into clear audio & vice versa.
CDLNumber	BYTE	Channel associated with CIU
DIUNumber	BYTE	Digital Interface Unit. Translates ASTRO clear secure data into analog audio & vice versa.
DBLNumber	BYTE	Channel associated with DIU

†Objectspace data types
Unused string fields are null. Unused number fields are set to zero

The version number is used to indicate the structure of data contained within the call record. In order to maintain compatibility with future versions, changes to call record structures will be performed in an additive nature. That is, current members of the call record will not change in position, size, or meaning. Each call record will contain a list to store participant information. There will be at least two participants in a call record; the calling and called parties. Any additional connections that are conferenced in or transferred to are appended to the end of this list. Only one active VOX and CTI based Master Call Record is allowed per recording input channel at any given time.

CRG Software Architecture

FIG. 18 shows the processing threads and data structures that comprise the CRG module in a preferred embodiment.

Event Processing: when the CRG is created and initialized, three threads are created. These threads are the CRG Event Processor thread 1810, Facade thread (The terms “facade,” “facade,” and “fascade” are used interchangeably in this disclosure) 1812 and Local Data Store thread 1816. Additionally, three message queues are created and are known as the Recorder 1824, Facade 1832, and Data Store 1844 queues, respectively. These queues enable the processing of various input messages in a de-coupled fashion within the CRG, so that any delay in one area of communications does not affect the processing of another area. Each thread is described below.

Event Processor Thread: the Event Processor is the primary thread of the CRG module. Its responsibilities include reading any messages placed in the Recorder 1824 and

US 6,785,370 B2

47

Facade 1832 queues. The processing activities that occur in response to these messages cause updates to be made to call records belonging to one of the recording input channels 1856. If these changes cause a call record to be completed, a message is sent to the Date Store queue 1844 requesting that the call record be made persistent in the local database. This thread is also responsible for processing state change messages, that cause memory resident structures to be refreshed or to shut down the CRG module.

Facade Thread: The Facade thread handles messages that come from outside the Voice Server. Its primary function is to look for messages placed in the CRG's external Microsoft Message Queue (MSMQ) 1864 where events may arrive from other components within the overall subject system. Upon receipt of a message, the Facade thread reads the message, translates it into an appropriate format for the CRG's internal data structures, and places the translated copy in the Facade Queue 1832. This thread is known as the Facade, because it manages the external interactions of the CRG with the other components within the subject system.

Local Data Store Thread: The Local Data Store thread 1816 processes requests from the CRG Event Processor thread 1810. The primary purpose of the Local Data Store thread 1816 is to take internal Master Call Record (MCR) structures and translate their contents into structures compatible with database technologies, such as Microsoft SQL Server, or comparable types of storage means. These resultant structures are stored within the database in order to make the call record persistent.

Characteristics of some switches mandate that the CRG be able to handle CTI events that are not real-time. Some switches batch events and send them out periodically. CRG configuration settings that limit the history list by time must be set long enough to accommodate the switch characteristics. Therefore, call records that are generated between switch reports (via recorder events) will not be finalized until a configurable time period (window) after which the call record terminated. This window (CallHoldoverPeriod) needs to be set to a minimum of the period of time between switch reports. Once a call record leaves this time window, it is marked as read-only and committed to the local data store.

A situation that must be dealt with is when the telephone switch is not time synchronized with the rest of the system. To facilitate the merger of recorder and switch events effectively in non-time-synchronized systems, alternate embodiments of the subject system are described.

One alternate embodiment of the subject system has a mechanism that synchronizes the clocks in the system (manually or automatically) on a periodic basis. This must guarantee time skews of less than some small and known quantity. A second embodiment has a mechanism for measuring the time delta between the switch and the subject system. This value is updated periodically and used by the CRG during the merging process. A third embodiment implements a combination of the first two.

During the call record merging process, a global time delta is used to adjust switch event time stamps before comparing to existing call record data.

The following paragraphs define the types of events the CRG is designed to accept and process. These events may cause the CRG to initialize, process metadata into call records, or prepare the system for shutdown.

The Master Controller (a sub-component of the present system's Scheduling & Control Services) supplies system events. The Master Controller notifies the CRG of system

48

related changes such as configuration changes, CTI server status and system shutdown events. The CRG changes its behavior based upon events received from the Master Controller.

System Events: The CRG provides an interface that allows the client application to control its states of operation. This is accomplished with an interface class that is used by most system components in the subject system. The interface is named IProcCtrl and supports the following methods: Initialize(); Start(); Stop(); Pause(); Resume(); Ping(); and Shutdown().

In addition to these methods, the CRG supports two event messages that inform it of status changes that are needed to either update its memory resident configuration information or change its mode of operation. These methods are CtiStatus and AgentExtensionStatus. Each method is described in the following paragraphs.

Initialization Event: This method is the first method that should be called after the CRG has been created. When the CRG object is created, it retrieves configuration information from the subject system's database. This information describes the number of channels in the recorder, the switch location where each channel is connected, any fixed associations of telephone extensions or agent identifiers. Also included are parameters that determine the behavior of the CRG. Threads are spawned to handle the processing of CRG events, communicating with external metadata contributors, and processing information into the Call Records tables. These threads are created in a suspended state and require the Start or Resume commands to begin processing activity.

Start Event: This method should be called after the Initialization event. It resumes all threads of the CRG enabling it to process incoming events.

Pause Event: This method suspends all threads of the CRG.

Resume Event: This method is called after the Pause command to enable all CRG threads to continue processing.

Ping Event: This method is used by client applications to test the connection to the CRG. The method simply returns a positive acknowledgment to let the client know that the CRG is still running.

Shutdown Event: This method notifies the CRG when the subject system is shutting down so that it can cleanly terminate itself. The shutdown event supports a single parameter (ShutdownMode) that indicates how it should shutdown.

If the ShutdownMode is specified as "Normal", all pending events read from the input event queues and processed into the call records, any open call records remaining are closed at the current time and written to the database.

If the ShutdownMode is "Immediate", input event queues are cleared without processing into call records, open call records are closed and written to the database.

Once these actions are completed, the CRG threads terminate. At this point, it is now safe for the client application to release the resources of the CRG.

Stop Event: This method is implemented for consistency with the common interface of IProcCtrl. The CRG has no purpose for this method and just returns a positive acknowledgment.

CtiStatus Event: This event informs the CRG of the operational status of the CTI server that is providing it with telephony metadata needed for CTI call record generation. The Scheduler component of the subject system is responsible for maintaining a heartbeat with the CTI server to

US 6,785,370 B2

49

detect when connection has been lost. Any changes in CTI server status result in a CtiStatus message directed at the CRG.

This message contains one parameter that indicates the new state of the CTI Server. If the parameter indicates that a CTI Server has become operational, recording input channels associated with the CTI Server change from "Degraded" mode of operation of "Normal" mode. If the parameter indicates that the CTI Server is not operational, recording input channels associated with the CTI Server change from "Normal" mode of operation to "Degraded" mode.

AgentExtensionStatus Event: This event indicates that a change in one of the Agent or Extension tables has occurred. Since the CRG uses these tables to associate with recorder channels, the memory resident version must be updated. Therefore, this event causes the CRG to read these tables and update its memory resident copy.

Call Record Events: When a call record event is received, the message is interpreted to determine which recording input channel may be affected. Any filtering necessary on a per channel basis is performed at this stage. Call record events are then dispatched to the appropriate Call Record

50

Channel Manager. There is a separate call record channel manager, which is a software sub-component of the CRG, for each recording input channel in a Voice Server. There are three messages that directly contribute to the creation and completion of call records. One comes from the CTI Server in the form of a CTI Event. The other two originate from the recorder and are the VoxSummary and VoxDisconnect messages. Each message is described in detail below.

CTI Event: The CTI Event is a message originating from the CTI Server software module that processes the information received from the telephone switch. The message details each participant involved with the phone call as well as information global to the call such as ring duration and DTMF codes. A CTI event message is sent to the CRG whenever a change in participant status occurs as well as when new ones enter the call. The messages are cumulative in that all information of the previous messages is contained in the new one with any additions included. This makes for a more robust system in cases where one of the messages is lost.

The pseudo code for processing a CTI event is shown below:

```

Pseudo code for CTI Event
//-----CTI Event (BEGIN)-----
// Don't process CTI events if we're not in correct mode
Is this recorder channel configured to receive CTI event data?
{
  // Yes
  Does this event match an MCR in my CTI History list?
  {
    // Yes
    Update MCR participants with matching one in CTI event
    Add any new participants to MCR.
    UpdateMediaFiles() (see pseudo code)
  }
  // End - Does this event match an MCR in my history list?
  Otherwise
  {
    Create new MCR
    Initialize MCR Start time from Oldest Participant Start time in event
    Copy participants from event to message to MCR.
    // Now that we've updated the participants, see if
    // we need to change media file associations.
    UpdateMediaFiles() (see pseudo code)
    Insert new MCR Into Cti MCR history list.
  }
}
Are there any participants still active?
Mark MCR as active
Otherwise
Mark MCR as complete
}
// End - Is this recorder channel configured to receive CTI event data?
//-----CTI Event (END)-----
//-----UpdateMediaFiles (BEGIN)-----
for each Recorder Participant in the MCR
{
  Is this not a new Recorder Participant?
  {
    //This participants start and/or end time may have been adjusted.
    //See if audio previously absorbed by it has to be returned to the VOX history
    list
    FindGiveBackMediaFiles() (see pseudo code below)
  }
  for each MCR in VOX History list with a time range that overlaps with this
  recorder participant
  {
    for each media file in this VOX MCR that's timespan overlaps with this
    recorder participant
    {
      CheckAndApplyMediaFile() (see pseudo code)
    }
  }
  //End - for each media file in this VOX MCR that's timespan overlaps with this
  recorder
  participant
  Did we consume all audio in this VOX MCR?
}

```

US 6,785,370 B2

51

52

-continued

```

    Remove VOX MCR from History list and delete it.
  } End - for each MCR in VOX History list that's time overlaps with this recorder
  participant
} End - for each Recorder Participants in the MCR
GiveBackAudio() (see pseudo code)
//-----UpdateMediaFiles (END)-----
//-----FindGiveBackMediaFiles (BEGIN)-----
for each media file associated with the given CTI MCR
{
    Was this media file contributed from the given recorder participant?
    { // Yes
        if media file lies completely outside recorder participant timespan?
        |<-----Participant Timespan----->|
                                     |<-----Media File timespan----->|

            Move this entire media file to the Giveback list
            Otherwise, If media file start time is before the recorder participants start
            time?
            |<-----Participant Timespan----->|
            |<-----Media File Timespan----->|
                Make a copy of this media file and set its end time to the participants start
                time.
                Add media file to giveback list.
                Set original media files start time to that of recorder participant.
            Otherwise, if media file end time is after the recorder participants end time?
            |<-----Participant Timespan----->|
            |<-----Media File Timespan----->|
                Make a copy of this media file set its start time to the participants end
                time.
                Add media file to giveback list.
                Set original media files end time to that of recorder participant.
            }
    } End - for each media file associated with the given CTI MCR
    //-----FindGiveBackMediaFiles (END)-----
    //-----GivebackAudio (BEGIN)-----
    // Sweep through VOX MCRs re-populating any giveback audio
    for all audio portions in given back list
    {
        if we find the VOX MCR this audio originally came from?
        { // Yes
            Attempt to merge the give back media file with an existing VOX MCR media
            file
                that's start or end time is adjacent to this ones.
            Otherwise, associate this media file with the VOX MCR.
        } End - if we find the VOX MCR this audio originally came from?
        Otherwise
        {
            // Original VOX MCR containing this audio file doesn't exist anymore.
            Create a new MCR.
            Associate the giveback media file with the new MCR
            Insert MCR into VOX History list
        }
    } End - for all audio portions given back
    //-----GivebackAudio (END)-----
    //-----CheckAndApplyMediaFile (Begin)
    Does Recorder Participant span the entire media file?
    |<-----Participant Timespan----->|
                                     |<-----Media File timespan----->|
        Move media file from VOX MCR to Cti MCR.
    Otherwise, Does Recorder Participant overlap with media file start time?
    |<-----Participant Timespan----->|
                                     |<-----Media File timespan----->|
        Make a copy of this media file and set its end time to the participants end time and
        associate with recorder participants MCR.
        Set the original media files start time to the recorder participants end time.
    Otherwise, Does Recorder Participant overlap with media files end time?
    |<-----Participant Timespan----->|
    |<-----Media File timespan----->|
        Make a copy of this media file and set its start time to the participants start time and
        Make another copy of this media file and set its start time to the participants end
        time and associate with VOX MCR.
        Set the original media files end time to the recorder participants start time.
    //-----CheckAndApplyMediaFile (End)

```

US 6,785,370 B2

53

VOX Summary Event: The VOX Summary Event is a message originating from the recorder associated with this CRG. It can be used in one of two ways.

The primary use of this message is to indicate the start of audio activity in real-time. When used in this mode, the VOXSummary command indicates the beginning of audio activity. But since the activity is not complete, the end time is set to indicate that the VOX segment is incomplete. The end time of incomplete media file is also set in this way. In this case, a VOX Disconnect message is required to complete the end times.

The second mode is used to indicate a history of audio activity. The VOX Summary start and end times reflect the period of time covered by all accompanying media files. The media files also have there respective start and end times filled in. This message is complete and thus requires no follow up messages. The VOXSummary message is shown below.

VOX Summary Message Format

Field Name	Description
Channel	Recorder channel of audio activity
VOXCrNum	Sequence number used to correlate related VOX events.
StartTime	time at which audio activity first started
EndTime	Time at which last audio activity ended.
Media Files	list of multimedia filenames used to store data with respect to this call record. (see below for details)
RingLength	Time from start of ring to off hook (in sec)
DtmfCodes	String of DTMF codes detected during VOXSummary period
ConnectReason	Indication of why VOX segment was started
DisconnectReason	Indication of why VOX segment was terminated
Media File Structure	
FileStartTime	Time corresponding to first byte of audio data in a file.
StartTime	Time corresponding to first byte of audio at which activity occurred
EndTime	Time corresponding to last byte of audio at which activity occurred
FileName	String containing name of audio file.
PathName	String describing the location of audio file.
iAssocPart	Used by the CRG to indicate with which Recorder Participant this audio segment is associated, when it is part of a CTI-based MCR.
dwVOXCrNum	Used by CRG to indicate which MCR in the VOX History list this audio segment originated.

The pseudo code for processing a VOX Summary event is shown below.

```
//-----VOXSummary (BEGIN)-----
Is this recorder channel configured to receive CTI event data?
{ // Yes
    // Attempt to merge media files in message with CTI based
    MCRs
    for each CTI MCR in History list
    {
        If any of the given media files fall inside the timespan
        of the Cti MCR?
        { // Yes
            // Merge media files with overlapping recorder
            participants in CTI MCR
            for each given media file in VOX Summary message
            {
                for each recorder participant in the given CTI MCR.
                {
                    CheckAndApplyMediaFile () (see psuedo code)
                } End - for each recorder participant in the given
                CTI MCR
            } End - for each given media file
        }
    }
}
```

54

-continued

```
} End - If any of the given media files fall inside the
timespan of the Cti MCR?
} End - for each CTI MCR in History list
Remove media files from VOX Summary message that are
completely consumed
}
Any unabsorbed audio remaining in message?
{ // Yes
    Create MCR for remainder of audio.
    Insert MCR into VOX History List
}
//-----VOX Summary (END)-----
```

VOX Disconnect Event: The VOX Disconnect Event is a message originating from the recorder associated with this CRG. It is used to terminate a VOX segment that has been started by a real-time VOXSummary message.

The VOXDisconnect message is shown below.

VOX Disconnect Message Format

Field Name	Description
Channel	Recorder channel of audio activity
VOXCrNum	Sequence number used to correlate related VOX events.
Time	End time of the VOX segment. Also indicates the end time of open media file.
DisconnectReason	Indication of why VOX segment was terminated

The pseudo code for processing a VOX Disconnect event is shown below.

```
// -----VOXDisconnect (BEGIN) -----
Is there a MCR in VOX History list with the same sequence
```

-continued

```
number?
{
    // Yes
    // Close and update all media files in both VOX and
    // MCR list related to this one
    Close Active media file in VOX MCR at given message time
    UpdateFromMediaFile ()
    // Update any CiMCRs that absorbed the audio file
closed.
    for each MCR in CTI History list
    {
        // Attempt to merge audio with MCR.
        // Look for matches with audio filenames.
        for each media file in Ci MCR contributed by this
        VOX clip
        {
            Close media file at given message time
            // Now that we've closed it, does this media file
            still
            // belong with this CiMCR?
            Does media file still fall in time span of MCR?
            { // Yes
                CheckAndApplyMediaFile () (see pseudo code)
            }
            Otherwise
            {
                Remove media file from MCR list and discard
            }
        }
        } End - for each media file in MCR contributed by
        this VOX clip
    }
    Close VOX MCR and mark as complete
}
// ----- VOXDisconnect (END) -----
// ----- UpdateFromMediaFile (BEGIN) --
// -----
    // Look for matches with audio filenames
    for each media file in MCR contributed by this VOX clip
    {
        Close media file at given message time
        // Now that we've closed it, does this media file
        still
        // belong with this CiMCR?
        Does media file still fall in timespan of MCR?
        { // No
            CheckAndApplyMediaFile () (see psuedo code)
        }
        Otherwise
        {
            Remove media file from MCR list and discard
        }
    }
    } End - for each media file in MCR contributed by this
    VOX clip
// ----- UpdateFromMediaFile (END) ---
// -----
```

Data Events: Data events are appended to the currently open associated call record. For CTI data events, this pertains to a currently open MCR based upon CTI connect events and containing a matching call record ID. For VOX data events, the currently open VOX call record is affected. If an open call record doesn't exist, an error condition is reported.

Correction Events: Correction events exist to remove a previous alteration to a call record after it has already been populated. One reason for such an event is to support selective record. An audio file that cannot be recorded due to customer or legal reasons might need to be removed from the call record or the entire call record might need to be deleted. The VOX event for a filename might have already been processed into a call record before the selective record mechanism has determined it not to be recorded.

Selective Record (Exclusion): Selective Record is an important feature of the subject system, imposed by customer requirements. If the customer does not want certain

participants recorded when they become involved in a recorded call, the CRG must exclude any audio associated with the call record for that participants' time of involvement. Implementing this feature is complicated by the varying characteristics of customer switches. If the telephone switch environments report events in real-time, recording of media can be prevented by turning the recording input channel off during the selective record participants' time of involvement. However, what happens when events are not reported in real time from the switch? The answer lies in the sweeping action of the CRG previously discussed for recorder participants.

The CTI Event message is routed through the Scheduler, and is altered by the Scheduler to indicate which participants re recorder participants as well as which ones are selective record participants. Recorder participants trigger the CRG to sweep any audio from VOX MCRs that overlap in time. When the CRG detects an overlap between recorder participant and selective record participant times, the audio that is swept into the CTI MCR for this overlap period is discarded. This causes the audio to be removed from both VOX and CTI MCRs, which prevents any chance of the audio being made available for playback or archive.

Selective Record Event: The Selective Record command is an event originating from the Scheduler. It identifies either a participant that is not to be recorded or that an entire call record should not be recorded. In one embodiment the system is capable of handling recording exceptions based upon information obtained from the CTI data. Criteria for selective record processing are discussed below.

Selective Record feature can take on two meanings. In one instance, a customer may want to record all telephony events except for ones that meet specific criteria. In a second instance, a customer may only want to record calls that meet certain criteria.

Since selective recording can possibly be triggered from multiple sources, in a preferred embodiment this decision process is located in the Master Controller, a sub-component of the subject system's Scheduling & Control Services.

Suggested reasons for not recording all or parts of a call are based upon the following examples of CTI event data.

Event Data	Explanation	Results
Agent exclusion based upon participants AgentID	Supervisor involved calls not to be included	Delete audio for agent's participation during the call, and the associated references in the MCR.
Exclusion based upon Extension or fully qualified phone Number of participant.	CEO involved calls not to be recorded. (whether at office or at home)	Delete audio for agent's participation during the call, and the associated references in the MCR.
Combination of AgentID of one participant and fully qualified phone number of another participant	Prisoner calls his lawyer.	Delete all audio as well as the entire call record.

Based upon these conditions and any future rules established inside the Master Controller (MC), exclusion can take place on audio recorded during a target participant's time of involvement or over the entire call record.

The chain of events involved in Selective Record (Call Exclusion) is as follows:

1. Recorder detects presence of audio and records to audio buffer.

US 6,785,370 B2

57

2. Recorder sends VOX events to CRG indicating presence of audio.
3. CRG creates new call record based upon VOX event.
4. The CTI server sends call events to CRG and MC.
5. CRG associates CTI event data with VOX based call record.
6. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a Selective Record (exclusion) command is sent to both Recorder and CRG indicating the start of the selective record interval.
7. Recorder deletes audio indicated in selective record message and continues to suppress recording until instructed otherwise.
8. CRG alters the call record to eliminate details of participant or deletes the call record.
9. Upon completion of the call, the CTI Server sends call events to the CRG and MC.
10. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a selective record (exclusion) is sent to the Recorder indicating the end of the selective record interval.
11. The Recorder resumes its normal mode of audio recording.

Selective Record (Call Inclusion)

1. The CTI server sends call events to CRG and MC. CRG creates MCR and populates with events. Since default is set not to record, the flag `m_bDontArchive` is set to prevent the local data store from writing it to the database.
 2. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a Selective Record (inclusion) command is sent to both Recorder and CRG indicating the start of the selective record interval. CRG sets `m_bDontArchive` to false and immediately instructs local data store to archive.
 3. Recorder detects presence of audio and records to audio buffer.
 4. Recorder sends history of VOX events to CRG in a `VoxSummary` message.
 5. CRG creates new call record based upon VOX event.
 6. CRG associates CTI event data with VOX based call record.
 7. Upon completion of the call, the CTI Server sends call events to the CRG and MC.
 8. MC checks for selective record triggers based upon criteria indicated above. If a criterion is met, a selective record (inclusion) command is sent to the Recorder indicating the end of the selective record interval.
 9. The Recorder resumes its normal mode of suppressing the audio recording.
- The format of the Recorder's Selective Record command is shown below.

Name	Type	Description
StartTime	time_and_date	Start Time of recording interval
EndTime	Time_and_date	End Time of recording interval

58

-continued

Name	Type	Description
bRecordAudio	bool	If true, record audio during the indicated interval. If false, suppress any audio recording during the indicated interval.

Since the recorder has no knowledge of participants or call record boundaries, the MC needs to inform the recorder when to start a selective record interval and when to stop. The boolean `bRecordAudio` signifies what action should be taken during this interval.

When an event occurs that triggers the start of a selective record interval, the Recorder's selective record command informs the recorder of the interval start. The End time is most likely not known at this point so it is set to some invalid value in order to indicate that audio should be recorded (or suppressed) for an indefinite period until a subsequent command is received.

When an event occurs that triggers the end of a selective record interval, the recorder's selective record command informs the recorder of the interval end. The End time indicates when the selective record interval is complete. The recorder returns to its normal recording mode based upon its original configuration.

Any selected audio committed to file needs to be removed from the file and replaced with a silence entry for that period.

The format of the CRG selective record command is shown below.

Name	Type	Description
MCR Number	UUID	MCR affected by this selective record command
Participant Index	UINT	Index of participant in MCR not to be recorded. (if Reason = 1)
Reason	BYTE	1 = Participant, 2 = Entire Call

For the CRG, only a single event that indicates what is selectively recorded is needed. If the Reason code indicates that the entire call record is to be deleted, the CRG will mark the call record such that it is removed from the database if it has already been written or not logged in the first place. If selective record affects a specific participant, the call record can either be left unmodified (since the recorder has already handled deletion of audio) or the participant can be over-written to remove his/her details.

The system configuration can be adjusted so that the CRG will operate in either fashion, depending on whether removing the audio alone is sufficient for the desired application of the system, or if the metadata must also be removed to eliminate the records of telephone numbers dialed, etc.

CRG Software Implementation

In the preferred embodiment of the subject system, the CRG is implemented as an in-process COM DLL that is associated with the Audio Recorder process, and therefore these two components reside together upon the Voice Server. COM, here, is Common Object Model, a distributed computing architecture designed by Microsoft Corporation to facilitate cooperative processing among software elements on a LAN. DLL is Dynamic Link Library, a means whereby executable code can be encapsulated in a package that can

be loaded upon demand and shared by several programs, rather than being packaged as a separate, isolated executable program. The Audio Recorder process is responsible for creating the CRG COM object as well as starting and stopping the CRG subsystem. The Data Store module that interfaces with the CRG is a statically linked DLL.

Class Design

FIG. 19 illustrates the class diagram of the Call Record Generator. The CRG module is itself comprised of a plurality of modules, as shown in the figure, and explained below.

CallRecordEvent Processor—the CallRecordEventProcessor class 1912 is the main class of the CRG. It is instantiated during the Initialize method call of the CRG interface. It is responsible for allocating the rest of the CRG objects. On instantiation, it acquires the channel count for the recorder (currently limited to 128) and instantiates a group a classes for each recording input channel. These classes include a CallRecordChannelManager 1916 and RecorderLocation 1920 for each channel. The CallRecordEventProcessor 1912 creates the Recorder 1924 and Facade 1928 Event input queues. Reading and processing of configuration information from the subject system's database takes place in the CallRecordEventProcessor 1912. Events received that cause a change in configuration are processed there.

CallRecordChannelManager—This class manages the call records for a specific recording input channel. It is responsible for creating, populating, and closing call records with event information received from the CRG event processor. If event information is deemed as significant, the CallRecordChannelManager 1916 will send an event to the DataStoreEventQueue 1932 in order for the update to be reflected in the local data store.

MasterCallRecord—This class 1936 holds information that is global to an entire call. Global information includes identifiers for the call record, the start and stop times of the entire call, the recorder location with respect to the switch, and flags indicating the call record status. It also contains a list of the participants within a call, based upon information supplied by CTI events. It acts as a centralized point of control for merging call record information for a given telephone call.

VoxCallRecord—This class 1940 is a superclass of the MasterCallRecord class 1936. It contains information dealing with events provided by the recorder. It holds the details of a call, such as the start/stop times, media filenames and other data that can be supplied by the recorder.

RecorderLocation—This class 1920 holds the information relating a logical device on a telephony switch with a specific Voice Server and recording input channel. The following table indicates configuration information needed by the CRG at runtime.

Configuration Field	Type	Acceptable Values	Default	Description
sSysTimeCoupling	String	"TIGHT", "LOOSE"	"LOOSE"	Indicates how time-based recorder and CTI events are compared to determine a match. TIGHT - Recorder times must fit entirely inside CTI times for a positive result. LOOSE - Recorder times need to overlap with CTI times for a positive result.
nCompleteCallHoldOver Period	DWORD	0.42949672 96 (in sec)	90 - For realtime CTI. (Much larger if non- realtime CTI)	Maximum number of seconds that a completed call record is kept in the history list. This holdover allows events coming from different sources to affect the call record before it is made persistent. After this holdover period expires, no more events can update the call record.
nActiveCallHoldoverPeriod	DWORD	0.42949672 96 (in sec)	86400 (24 hours)	The maximum number of seconds a call is allowed to exist before being forcibly closed. This is used as a safeguard against missing CTI or Recorder events that would normally end a call record.
nMCRMaxSize	WORD	0.65535 (in entries)	100	Maximum number of entries allowed in the MasterCall Record history list
nSystemSkew	WORD	0.65535 (in sec)	0	A known, fixed difference (in seconds) that specifies the skew between a Recorder clock and a PEX clock. Used to adjust incoming CTI event times before processing and comparing with Recorder event times.
ynCTIDataFromRecorder	bool	1 = yes 0 = no	yes	Identifies, in cases where Recorder and CTI information overlaps, which source is preferred to populate the call records.
nSaveVoxClipsLonger ThanSeconds	WORD	0.65535	6	This setting is used to avoid creating VOX based call records

US 6,785,370 B2

61

62

-continued

Configuration Field	Type	Acceptable Values	Default	Description
				from noise on recording input channels. It directs the CRG to discard any VOX clips that do not exceed the specified number of seconds in duration.

Stream Control Manager

As noted above, in a preferred embodiment, the system of the present invention taps into activity on a PBX (Private Branch Exchange) by intercepting audio on either the trunk or extension side of a phone call. The tapped audio is then redirected as input to a channel on a DSP (Digital Signal Processor) based voice processing board, which in turn is digitized and stored into program-addressable buffers. The recorded audio is then combined with descriptive information ("metadata") obtained through a Computer Telephony Integration (CTI) communications link with the PBX and stored as a single manageable unit ("Voicedata") to facilitate its subsequent search and retrieval.

The preferred embodiment leverages Computer Telephony Integration, to supplement the recorded audio data. As discussed above, CTI is provided through a data link from specific telephone switching equipment located at the customer site, which is then input to the recording system's CTI Server. Supplied data includes such items as telephone numbers of involved parties, caller ID/ANI information, DNIS information, and agent ID numbers. The CTI Server performs the task of analyzing and reorganizing data from both the real-time and SMDR (asynchronous) links, and passing the results onwards to the remainder of the recording system for further processing.

A module called the "Call Record Generator," or CRG, discussed above, is then responsible for collecting data from the CTI Server, creating 'master call records' and attempting to match those records with existing recorded audio data. If the CRG receives CTI information indicating that audio data recorded on two Voice Servers is related (for example, due to a transferred call), records will be generated for each portion with a common call record ID. This ID can later be used to query for all the pieces (or "segments") comprising the complete call. In addition, each segment will indicate the Voice Server which contains that piece of the call.

During playback, the User Workstation's player module connects to a program located on a Voice Server called the Playback Server, or PBServer. The machine name of the particular Voice Server with which a communications session should be established, stored by the CRG in the call record table of the Voicedata storage module, is passed into the player module after being extracted by the User Workstation's call record browser. A call record playback request is then submitted, which causes the PBServer to query for a specific call record's audio files located on that physical machine, open them, and prepare to stream the audio upon buffer requests back to the client. If successful, a series of requests is then issued from the client, each of which will obtain just enough audio to play to a waveOut device while maintaining a safety net of extra audio in case of network delays. Upon a request to "Move" within the scope of a call record, the PBServer will reposition its read pointer to the desired location and then begin passing back buffers from that point. This series of Request and Move commands will

continue until the user chooses to end the session by shutting down the client-side audio player.

When a call is transferred between locations, it is possible that the call may span multiple Voice Servers, since the extensions or trunks involved may be monitored by different recorders. If this is the case, the audio data is spread out between playback servers, and it must be properly pieced back together to reconstruct the complete call for a playback client.

There are several possible solutions to the problem. First of all, one could choose one central server and copy in all data from the involved servers. This is as slow as copying the files locally to the client, but it at least consolidates the data to one location for the playback server to operate on. Assuming that this method is chosen, however, several new problems arise. First is the issue of drive space: depending on the number of transfers and recorders involved with a call record, the central playback server could end up suddenly storing a large number of files. This is multiplied by the total number of clients requesting playback sessions. Soon enough, a large amount of unpredictable space is being allocated and freed without any reasonable way of estimating the space necessary to service all requests. Similarly, the processor and memory load on this server is taking the brunt of being used for every playback request, since even normal, single recorder playback sessions would be routed through this one machine.

Another solution would be to have the central playback server run some intermediate process that would stream all of the data from the multiple servers back to each client, like a "funnel." This would avoid the copying and drive space issues, but there are still two problems. First, the centralizing of this server once again puts the entire load on a single machine. But more importantly, if multiple streams are being funneled through this one location, the server would somehow need to organize the streams so that during playback, they appear to be arranged in the proper order.

The Stream Control Manager (SCM) used in accordance with a preferred embodiment is the result of addressing the issues referred to in the second solution discussed above. With regard to the resource issue, the solution was to simply move the "funneling" module from one central server to the client side. In this way, servers are still providing the actual requested data, but it becomes the client side's responsibility to bring the data together. Yet the SCM remains a separate, COM-based module so encapsulation is still maintained (a client application is not hard-wired directly into the SCM code). This was intentional since other system modules in alternate embodiments of the system need to reuse the SCM to gather playback data (e.g., for phone handset playback support instead of LAN playback support) or to gather audio from a multitude of Voice Servers for long-term offline storage on DAT or DVD media.

The process of stream management begins when the SCM is sent a list of segments which comprise the entire call. Each

US 6,785,370 B2

63

segment includes the machine name of the Voice Server, the segment's start time, duration, channel ID, and an event callback routine provided by the client which serves as a destination for the final organized data.

Once this list is received and stored as a vector (array), the SCM proceeds to try connecting to all servers required to play back this call. The connection, if successfully established, is associated with its respective segment via a pointer in the segment entry. The connection is also added to an array so that if a subsequent segment's server is the same as an earlier segment, the connection can be reused. This may occur if a call transfers away to a line monitored by a second recorder and is later transferred back again to the original line. If the process cannot complete successfully (i.e., if a Voice Server is malfunctioning), playback is aborted to avoid skipping over any necessary data.

Next, the SCM goes through its list of segments and for each, handshakes with its server through a series of function calls. During this phase, the SCM informs each playback server of the desired segment to stream back by providing its start time, duration, channel ID using the parameter data that was passed in earlier. Once again, if any part of the procedure fails, the entire initialization (and thus playback) is aborted. At the completion of this phase, every server should have loaded all the audio files associated with their portion of the entire data stream. Each is now ready for audio buffer requests.

The SCM then waits for a client to execute a "Start-Stream" call. In a graphical interface, this would occur, for example, when a user hits a Play button or begins a Save operation. Once this function is called, a separate thread spawns which will handle the entire process.

First, the current play position is checked to see which segment to begin playing on (a Move operation, explained below, controls the manual repositioning of this value). This is determined by looping through all of the segments, adding each segment's duration to a running total. When the current segment's duration added to the total exceeds the play position, that is the segment which contains the current play position.

Once this calculation is complete, a loop begins which starts from the previously determined segment and proceeds through the rest of the segment vector. For each segment, requests are formed for a predetermined buffer size and sent to the associated server. Once a buffer is returned, based on a flag configurable from the client, the SCM will either directly send back this data or "slice" it for the client first before returning it. Here, slicing refers to a process of dividing the buffer into smaller buffers by a least common multiple known as a block align; this is sometimes useful to a client with a graphical component because the interface may need to reflect the amount played in smaller subdivisions.

When it is detected that all data from a segment has been requested, the SCM automatically steps to the next segment (possibly located on a different Voice Server) and begins requesting data from it instead. Because all Voice Servers are pre-loaded with the data and "ready to go," this process takes place in a fraction of a second, and the client does not sense any gap in the audio data being returned. In fact, the only true method for discerning the segment boundaries involves listening for normal, audible indicators of a transfer being made (clicking, ringing, or hearing the voice of a new participant) as provided through the telephone switch environment.

At the close of a play session (e.g., the user hits Stop or Pause in a typical audio playback GUI displayed in con-

64

junction with the GUI described in FIG. 16) a StopStream call is made to the SCM. The thread in turn detects that the stopped state has been entered, exits from the request loop code, and frees up any used resources. Finally, it informs the client that a Stop event has occurred. If the entire call record is played without calling StopStream, the SCM performs the same exit and cleanup code, but informs the client that a Done event has occurred instead.

Movement within the overall stream is straightforward, given the aforementioned method that the SCM uses to determine which segment to begin playing from. A global variable holds the total number of milliseconds of audio data requested thus far. When a Move is performed, the server containing the data at the destination position is told to re-position itself, and the current play position is reset. Now, once StartStream executes again, it will initially start requesting from the server that was just moved to. And because that server had also moved its position pointer ahead, data will not be streamed from the beginning of the segment, but from where the Move position fell within that segment. Thus movement is a synchronized action completely transparent to the client, who is, ultimately, only interested in treating the data as a single stream.

SCM Pseudo-code

1. Initialize receives segment description data (start time, duration, etc.)
 - a.) Form a vector of all segments.
 - b.) Try to connect to all segments' servers.
 - c.) If there is an error connecting to any server, exit.
 - d.) Try to initialize each connected server.
 - e.) If there is an error initializing any server, exit.
 2. If StartStream received:
 - a.) Go through segment list. Find segment of current play position.
 - b.) Starting with that segment, contact the associated server and begin requesting buffers.
 - c.) If option set, divide up buffer into smaller chunks.
 - d.) Send buffer(s) to client via event callback.
 - e.) Repeat until all data requested for this segment on that server.
 - f.) Repeat from step b. with next segment in list.
 3. If Stop received:
 - a.) Exit from request loop.
 - b.) Clean up used resources.
 - c.) Send "Stop" event back to client.
 4. If Stop not received, but all data from all segments played:
 - a.) Exit from request loop.
 - b.) Clean up used resources.
 - c.) Send "Done" event back to client.
 5. Move received:
 - a.) Go through segment list. Find segment of desired play position.
 - b.) Contact the associated server and reposition to that desired position.
 - c.) Reset current play position variable to reflect change.
- Detailed flow diagrams describing SCM operation are provided in FIGS. 20, 20A, 20B, 21, 22, 22A, 22B, and 22C. FIG. 20 illustrates the initialization process of the Stream Control Manager. The Initialization Sequence begins when a user enters the User Workstation playback software and at step 2010 queries for a recorded call record by desired criteria. At step 2012 a call record browser displays resulting call records. At step 2014 the user selects the desired record

US 6,785,370 B2

65

for playback. At step 2016 the browser invokes a PbkControlWin object: a dialog containing the 'player' ActiveX control.

At step 2020 the browser sends information to PbkControlWin about all segments comprising the call record. If at step 2024 immediate playback is not required, at step 2028 the entry is added to a playlist for future playback, and at step 2030 SUCCESS is returned. If at step 2024 immediate playback is required, at step 2032 the call record ID and segment list are forwarded to a GUI Player module. At step 2038 (see FIG. 20A) the player module instantiates a local SCM (StreamControl) object and stores a pointer in m_plStreamControl. At step 2040 the player module accepts the data, displays starting time and total duration (by parsing out string data), and forwards it to the final module, the Stream Control Manager (SCM), for audio playback.

Step 2046 begins the creation of a segments vector. At step 2046, a segment is parsed out from segList. At step 2048, recorder ID, start time, duration, and channel are parsed out from the segment. At step 2050, a new SEGMENT structure is created from recorder ID, start time, duration, and channel. At step 2052, a new SEGMENT is added to the SEGMENT vector. At step 2054, if all segments have been parsed from segList, at step 2058 an element is gotten from the SEGMENT vector. If at step 2054 more segments remain to be parsed from segList, steps 2046, 2048, 2050, and 2052 are repeated.

After step 2058, the program determines at step 2060 whether a new DCOM connection is required to the recorder for this segment. If not, at step 2062 the existing pointer is copied from the Connections vector to the server pointer in the SEGMENT vector and the program proceeds to step 2076. If at step 2060 the connection is new, a connection is made to the indicated recorder's "PlayBackServer" DCOM object using CoCreateInstanceEx. At step 2066 the program checks whether the object instantiated successfully. If not, at step 2068 a log error message occurs and at step 2070 ERROR (C) is returned. If at step 2066 the object instantiated successfully, at step 2072 (see FIG. 20B) the new object's pointer is added to the Connections vector. At step 2074 the program determines whether all segments have been connected. If not, the program returns to step 2058. If at step 2074 all segments have been connected, at step 2076 an element is gotten from the SEGMENT vector. At step 2078 the program queries for a list of wave files on the server that go with this segment. At step 2080 the program determines whether the query was successful. If not, at step 2082 a log error message occurs, and at step 2084 ERROR (C) is returned.

If at step 2080 the query was successful, at step 2088 the program opens the wave files on the server and prepares them for streaming. It also returns the wave format of the audio in the segment. At step 2093 the program determines whether the wave files and format were obtained successfully. If not, at step 2094 a log error message occurs and at step 2095 ERROR (C) is returned. If step 2088 is determined at step 2093 to have been successful, at step 2096 the program checks whether all segments have been initialized. If not, the program returns to step 2076. If so, step 2097 is performed and at step 2098 SUCCESS is returned.

FIG. 21 illustrates how the program manages a Player Object 2110 and a PbkControlWin Object 2132.

FIG. 22 illustrates the playback sequence of the Stream Control Manager. Initially, at step 2202 a user has completed initialization and is waiting to hit Play in the Player GUI. At step 2204 the user hits the Play button. At step 2206 a message is sent to the Play method in the Player ActiveX

66

control. At step 2210 the Play method in Player ActiveX control causes the output buffers to be "sliced" to increase the number of smaller buffers sent, thus increasing the resolution of the "totalPlayed" variable. At step 2218 Play method causes the server-side position to move to the current slider position. At step 2222 the program gets segment i++ from the SEGMENT vector. At step 2224 (see FIG. 22A) the program determines whether the End Time offset for segment i is greater than curPosition. If not, the program returns to step 2222. If so, the program proceeds to step 2226 and causes the file pointer on the server side to change to the appropriate new location. The program checks at step 2230 whether step 2226 was successful. If not, at step 2232 a log error message occurs and at step 2234 ERROR (C) is returned.

If at step 2230 step 2226 is determined to have been successful, at step 2238 the program calls StreamControl::StartStream. At step 2242 the program gets segment i++ from the SEGMENT vector. At step 2244 the program calls CoMarshalInterThreadInterfaceInStream to marshal a DCOM pointer member across a thread boundary. At step 2246 the program determines whether all SEGMENT elements have been marshaled. If not, the program returns to step 2242. If so, at step 2248 the main SCM streaming thread is spawned.

FIG. 22B illustrates an SCM main streaming thread. When the thread begins, at step 2250 the thread gets a segment from the SEGMENT vector. At step 2252 CoCetInterfaceAndReleaseStream is called to unmarshal a DCOM pointer member across the thread boundary. At step 2254 the thread checks whether all SEGMENT elements have been unmarshaled. If not, the thread returns to step 2250. If at step 2250 all SEGMENT elements are determined to have been unmarshaled, at step 2256 the thread gets a segment from the SEGMENT vector. The thread then checks at step 2258 whether the End Time offset for segment i is greater than curAmountRequested. If not, the thread returns to step 2256. If so, at step 2260 the thread gets Segment[i++]. The thread checks at step 2262 whether i is less than the highest segment number. If not, an Event::Done method is called at step 2264, and at step 2266 SUCCESS (C) is returned. If so, at step 2268 the thread determines whether this is the first segment to be played in this instance of the thread. If not, at step 2270 the thread calls PServer::PositionPlay(totalRequested) for Segment[i] and goes to step 2272. If so, the thread goes directly to step 2272.

At step 2272, the thread checks whether totalRequested is less than Segment[i].endTimeOffset. If not, the thread returns to step 2260. If so, the thread proceeds to step 2274 and checks whether totalRequested plus bufferSize is less than or equal to Segment[i].endTimeOffset. If not, at step 2276 the thread calculates a new bufferSize in multiples of the audio format's "block align." and proceeds to step 2278 (see FIG. 22C). If so, the thread proceeds directly to step 2278. At step 2278, the thread calls PServer::ReqBuffer for Segment[i]. This is the core routine that actually retrieves a buffer of data from the PlayBack Server. At step 2286 the thread checks whether step 2278 was successful. If not, at step 2284 a log error message occurs, and at step 2282 ERROR (C) is returned.

If at step 2286 the thread determines that step 2278 was successful, at step 2287 totalRequested is set equal to totalRequested plus Actual returned buffer size. At step 2288, the thread checks whether Blockslicing is enabled. If not, at step 2289 the thread sends the buffer back to the Player via Event::SendData method and returns to step

US 6,785,370 B2

67

2274. If BlockSlicing has been enabled, at step 2292 the thread checks whether the CODEC is Dialogic OKI ADPCM or PCM. If not, at step 2293 the slice of the slices is set equal to the audio format's block align and the thread proceeds to step 2296. If so, at step 2294 the size of the slices is set to an even dividend of the buffer size (e.g., one-tenth of the buffer size). At step 2296, the thread copies out "slice size" from the buffer and sends it back to Player via Event::SendData method. At step 2298 the thread checks whether the entire buffer has been sent back. If not, the thread returns to step 2298. If so, the thread returns to step 2274.

The Stream Control Manager could theoretically be adapted to be used in more general streaming media situations, outside that of communications recording systems. In most current stream-based systems for network-based playback of audio content, such as RealMedia and NetShow, two general broadcast architectures exist known as unicast and multicast. Unicast involves a single client-server connection for data streaming, while in the multicast scenario a server pushes data to a single network address which multiple clients can then "tune in" to. However both models assume that data is being continuously fed from a single server. In the interest of load balancing, or if pieces of a streaming presentation were spread out across multiple locations, the SCM model could provide an innovative solution where the client side has the power to weave together many streams into a single playback session. An example could be imagined where a news organization, such as CNN, dynamically assembles a streaming broadcast for the online viewer from many different reports located on servers across the country. The components could be played seamlessly end-on-end using the SCM model, and if the viewer desired to rewind or fast-forward to a specific point in the stream, the SCM model would allow for complete transparent control.

The present invention is not to be limited in scope by the specific embodiments described herein. Indeed, modifications of the preferred embodiment in addition to those described herein will become apparent to those skilled in the art from the foregoing description and accompanying figures. Doubtless, numerous other embodiments can be conceived that would not depart from the teaching of the present invention, which scope is defined by the following claims.

All the features disclosed in this specification (including any accompanying claims, abstract, and drawings) may be replaced by alternative features serving the same, equivalent, or similar purpose, unless expressly stated otherwise. Thus, unless expressly stated otherwise each feature disclosed is one example only of a generic series of equivalent or similar features.

What is claimed is:

1. A method for constructing and maintaining data representations of lifetimes of telephone calls comprising one or more segments, audio data for each segment being recorded on one or more recorders, the method comprising:

- (a) constructing a call record for at least one telephone call;
- (b) receiving data regarding telephony events associated with one or more telephone calls;
- (c) matching a received telephony event with a constructed call record;
- (d) updating the matching call record based on the received telephony event data; and
- (e) combining the updated call record with data indicating the location of recorded audio data for the segment of

68

the call, to obtain a master call record representing the lifetime of the telephone call.

2. The method of claim 1 wherein the step of updating the matching call record comprises invoking one or more handler routines corresponding to the telephony event.

3. The method of claim 1 further comprising the step of translating the data regarding telephony events into a platform-specific format.

4. The method of claim 1 wherein in step (c) a confidence factor algorithm is used to determine whether a match has been found.

5. The method of claim 1 wherein the master call record comprises a serial number that identifies the telephone call.

6. The method of claim 1 wherein the call record is updated with data fields describing each participant of the telephone call.

7. The method of claim 5 wherein the call record is updated with data fields describing each participant of the telephone call.

8. The method of claim 1 further comprising the step of assembling and playing back segments of telephone calls using the recorder locations described in the master call record for each telephone call.

9. The method of claim 1 further comprising the step of using the master call record to display a graphical representation of said telephone call.

10. The method of claim 3 further comprising the step of using the master call record to display a graphical representation of said telephone call.

11. The method of claim 9 wherein the graphical representation comprises a representation of each segment of the telephone call.

12. The method of claim 9 further comprising the step of displaying a table comprising data from the master call record.

13. A computer program for constructing and maintaining data representations of lifetimes of telephone calls comprising one or more segments, audio data for each segment being recorded on one or more recorders, the computer program comprising:

- (a) software for constructing a call record at least one telephone call;
- (b) software for receiving data regarding telephony events associated with one or more telephone calls;
- (c) software for matching a received telephony event with a constructed call record;
- (d) software for updating the matching call record based on the received telephony event data; and
- (e) software for combining the updated call record with data indicating the location of recorded audio data for the segment of the call, to obtain a master call record representing the lifetime of the telephone call.

14. The program of claim 13 wherein software for updating the matching call record comprises one or more handler routines corresponding to the telephony event.

15. The program of claim 13 further comprising software for translating the data regarding telephony events into a platform-specific format.

16. The program of claim 13 wherein the software for matching a received telephony event with a call record uses a confidence factor algorithm.

17. The program of claim 13 wherein the master call record comprises a serial number that identifies the telephone call.

18. The program of claim 13 wherein the call record is updated with data fields describing each participant of the telephone call.

US 6,785,370 B2

69

19. The program of claim 13 further comprising software for assembling and playing back segments of telephone calls using the recorder locations described in the master call record for each telephone call.

20. The program of claim 13 further comprising software that uses the master call record to display a graphical representation of said telephone call.

21. The program of claim 19 further comprising software that uses the master call record to display a graphical representation of said telephone call.

22. The program of claim 20 wherein the graphical representation comprises a representation of each segment of the telephone call.

23. The program of claim 19 further comprising software for displaying a table comprising data from the master call record.

24. An article of manufacture storing a computer program for constructing and maintaining data representations of lifetimes of telephone calls comprising one or more segments, audio data for each segment being recorded on one or more recorders, the computer program comprising:

(a) software for constructing a call record for at least one telephone call;

(b) software for receiving data regarding telephony events associated with one or more telephone calls;

(c) software for matching a received telephony event with a constructed call record;

(d) software for updating the matching call record based on the received telephony event data; and

(e) software for combining the updated call record with data indicating the location of recorded audio data for the segment of the call, to obtain a master call record representing the lifetime of the telephone call.

25. The article of claim 24 wherein software for updating the matching call record comprises one of more handler routines corresponding to the telephony event.

26. The article of claim 24 wherein the computer program further comprises software for translating the data regarding telephony events into a platform-specific format.

70

27. A method for constructing and maintaining data representations of lifetimes of telephone calls comprising two or more segments, audio data for each segment being recorded on one or more recorders, the method comprising the steps of:

(a) constructing a call record for a telephone call comprising two or more segments;

(b) receiving data regarding one or more telephony events associated with the telephone call;

(c) matching said one or more received telephony events with said call record;

(d) updating said call record based on said received telephony event data; and

(e) combining said updated call record with data indicating one or more locations of recorded audio data for two or more segments of the call, to obtain a master call record representing the lifetime of said telephone call.

28. A method for constructing and maintaining data representations of lifetimes of telephone calls comprising two or more segments, the method comprising:

(a) constructing a call record for a telephone call comprising two or more segments, wherein at least one participant in said call participates in two or more of said segments;

(b) receiving data regarding one or more telephony events associated with the telephone call;

(c) matching said one or more received telephony events with said call record;

(d) updating said call record based on said received telephony event data; and

(e) combining said updated call record with data indicating one or more locations of recorded audio data for two or more segments of the call, to obtain a master call record representing the lifetime of said telephone call.

* * * * *

EXHIBIT H



US006870920B2

(12) **United States Patent**
Henits

(10) **Patent No.:** **US 6,870,920 B2**
(45) **Date of Patent:** **Mar. 22, 2005**

(54) **SYSTEM AND METHOD FOR MULTI-STAGE DATA LOGGING**

(75) Inventor: **John Henits, Bethel, CT (US)**

(73) Assignee: **Dictaphone Corp., Stratford, CT (US)**

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **10/187,865**

(22) Filed: **Jul. 2, 2002**

(65) **Prior Publication Data**

US 2002/0164007 A1 Nov. 7, 2002

Related U.S. Application Data

(62) Division of application No. 09/324,459, filed on Jun. 2, 1999.

(51) Int. Cl.⁷ **H04M 3/42; G11B 20/10; G11B 5/09; G06F 15/16**

(52) U.S. Cl. **379/207.02; 360/39; 709/217**

(58) Field of Search **379/207.02; 360/39; 360/5; 709/217, 218, 219; 714/4; 704/200; 705/39**

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,360,854 A	11/1982	Schergen et al.	361/149
5,127,003 A	6/1992	Doll, Jr. et al.	370/110.1
5,163,085 A	11/1992	Sweet et al.	379/89
5,195,128 A	3/1993	Knittl	379/67
5,199,062 A	3/1993	Von Meister et al.	379/67
5,274,738 A	12/1993	Daly et al.	395/2
5,392,329 A	2/1995	Adams et al.	
5,396,371 A *	3/1995	Henits et al.	360/5
5,448,420 A *	9/1995	Henits et al.	360/48
5,457,782 A	10/1995	Daly et al.	395/2
5,513,312 A	4/1996	Loebig	
5,535,261 A	7/1996	Brown et al.	
5,625,890 A	4/1997	Swift	455/67.1

5,710,978 A *	1/1998	Swift	455/67.11
5,819,005 A *	10/1998	Daly et al.	704/200
6,014,647 A *	1/2000	Nizzari et al.	705/39
6,122,239 A *	9/2000	Bodo et al.	711/111

FOREIGN PATENT DOCUMENTS

EP	0 642 250 A2	3/1995
EP	0 642 250 A3	1/1999
EP	0 978 983 A2	2/2000
EP	0 978 983 A3	1/2002
WO	WO 91/08572	6/1991
WO	WO 97/15007	4/1997
WO	WO 98/18283	4/1998
WO	WO 98/39901	9/1998
WO	WO 99/21336	4/1999

OTHER PUBLICATIONS

IEEE Communications Magazine, May 1997, "Interacting with Databases in the Global Information Infrastructure" by Tiziana Catarci.*

IEEE Journal, vol. 13, No. 8, Oct. 1995, "Security, Payment, and Privacy for Network Commerce" by B. Clifford Neuman.*

ASC Telecom GmbH, "System DL 2—Digital Voice Logging Unit on DAT—Cassette," Aug. 1992.

(List continued on next page.)

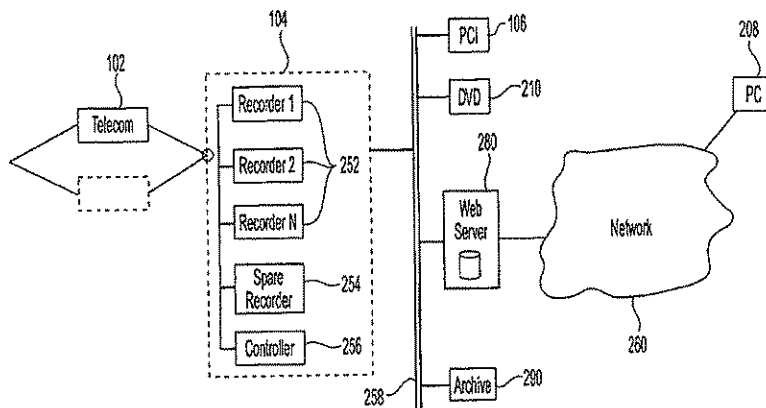
Primary Examiner—Benny Tieu

(74) Attorney, Agent, or Firm—Jones Day

(57) **ABSTRACT**

A multi-stage data logging system comprising a telecommunications stage for receiving, processing, and compressing data from one or more input channels, a recorder stage for storing said data to a memory device, a distribution stage for retrieving said stored data and distributing said data to one or more output channels, and a plurality of interface paths linking said three stages to one another. Different stages of the system can be located wide distances apart and the interface paths linking the three stages can be automatically switched to achieve fault tolerance of the system.

28 Claims, 5 Drawing Sheets



US 6,870,920 B2

Page 2

OTHER PUBLICATIONS

Dictaphone Corporation, "Series 9800 (DAT) Digital Logger—Operator's Manual," Nov. 1992.

Press Release, Dictaphone Corporation, "Digital Audio Tape Logger—Another Dictaphone First," Feb. 1992.

Secure Surveillance Systems, Ltd., "S³ DR 1000—16—32 Channel D.A.T. System".

Press Release, Dictaphone Corporation, "Dictaphone's New ProLog™ Digital Communications Recording System Provides Revolutionary Cost Savings/Playback Features," Jun. 1993.

Magnasync Corporation, Product Information and Brochures for Digital Voice Logger, Jul. 1992.

Racal Recorders, Inc., "Rapid Access Voice Logging Recorder," Oct. 1991.

Racal Recorders, Inc., "RAPIDAX Instant Recall Recorder," Jun. 1992.

Racal Recorders, Inc., "RAPIDAX Ranger Digital Tactical Logging System" Jun. 1995.

Racal Recorders, Inc., "RAPIDAX Ranger—Technical Specification" May 1994.

"Rapidax Ranger Sales Manual (Provisional)".

Nice, "Disk-Based Audio Storage/Retrieval Systems—DSN-1000".

"Voice Logging: Comverse Reports Initial Success for Its New Digital Voice Logging System," Edge, vol. 7, No. 209, p. 18, Jul. 1992.

Atis Assmann GmbH, Systemtechnics Division, "Multi-channel Monitoring and Recording—Overview," 1993.

Eyretel, Ltd., "Voice Recording Solutions".

Eyretel, Ltd., "E-500 16 Channel Digital Voice Recorder".

Eyretel, Ltd., "E1000 Digital Voice Recorder" brochure.

Eyretel, Ltd., "The E1000 Digital Voice Recorder" Information page, www.eyretel.com/E1000.htm, printed Apr. 1997.

Eyretel, Ltd., "An Outline of Eyretel and the E1000 Digital Recorder".

Eyretel, Ltd., Application Bulletin, "Networking Solutions for Eyretel Digital Voice Recorders".

Eyretel, Ltd., Application Bulletin, "Buyers Guide to Digital Voice Recorders for Financial Applications".

Nice Systems Ltd., "Nice-Log for Financial Institutions," brochure, 2 pages.

European Patent Application 0 550 273 A2 (corresponding to reference AK listed above), Dec. 31, 1991.

Patent Abstracts of Japan, Application No. 01267499, Oct. 13, 1989.

"Eventide's Digital Voice Logger," Teleconnect, Jun. 1991, p. 42.

"Questions and Answers: The Eventide VR240 Digital Audio-Logger," Eventide Inc., Mar. 27, 1991.

Racal Recorders, *Wordsafe*, Racal Recorders Inc., Pub. No. 3115-2, Dec. 1990.

TEAC Communications Recorders, CR-320/CR-310, TEAC America, Inc., Aug. 1994.

VR240 Manual, Eventide Inc., Jan. 6, 1992.

Magnasync/Comverse, "DVL 1000" brochure 2 pages.

Magnasync Corp., DVL 1000 Digital Voice Logger brochure, 2 pages.

Comverse/Magnasync, Digital Voice Recording DVL 1000 brochure, 1 page.

Comverse Technology, Inc., Audio Disk Observer brochure, 3 pages, 1991.

Nice Systems, Inc., "Customer Profile: Legal & General," brochure, 2 pages.

Eyretel, Ltd., "Digital Interfacing," brochure, 2 pages.

Sel-Tronics, Inc., "Series E-1000—The Intelligent Recorder," brochure, 2 pages.

Racal Recorders, Inc., "Racal Adds Remote 'Replay Over LAN' to Wordnet Voice-Logging Recorder," Jun. 1996.

Comverse Information Systems, "Section 1: Ultra 3000 System," 2 pages, Jun. 1995.

Siemens, "Data Voice: Digital Voice and Data Memory for Convenient Information Logging," Jan. 1994.

Nice Systems Ltd., "Company Profile," brochure.

Eyretel, Ltd., "Networking," brochure.

PCT International Preliminary Examination Report for PCT. App. No. PCT/US00/15419, mailed Aug. 22, 2002.

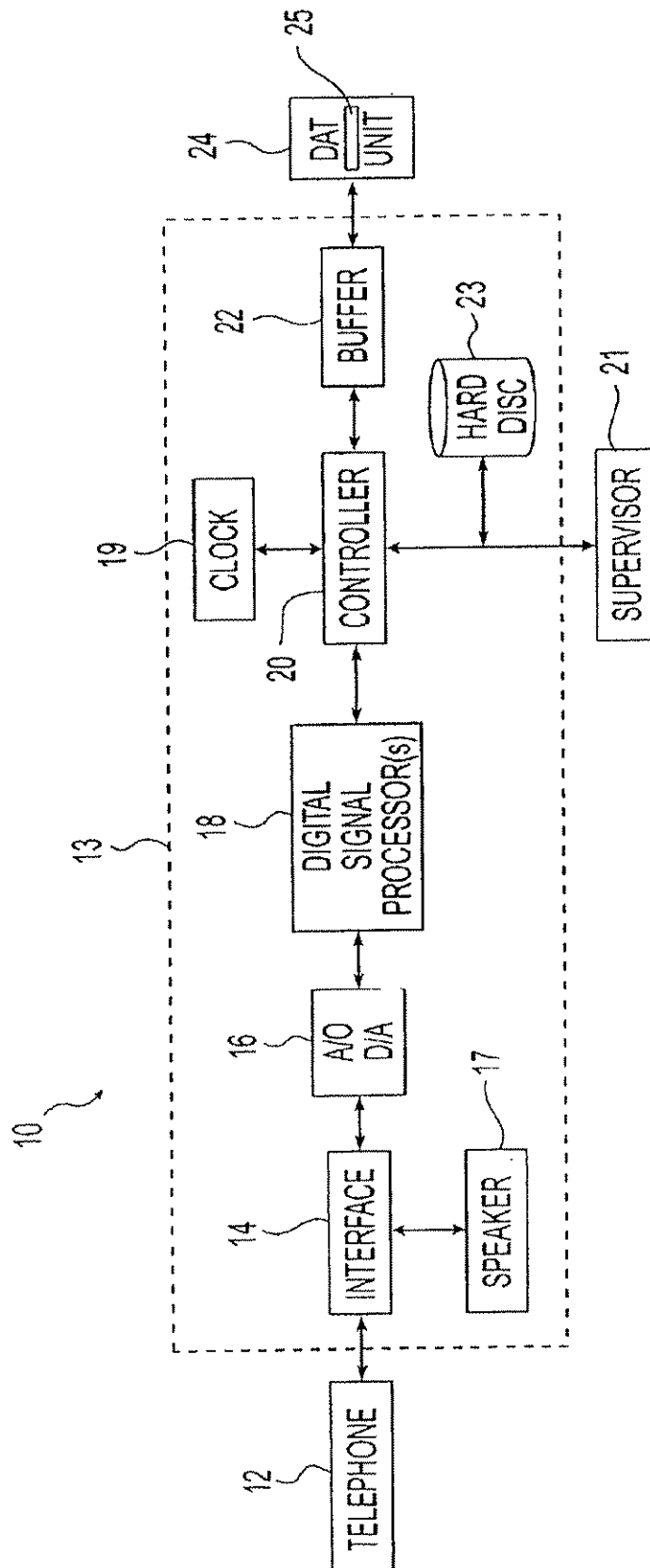
Supplementary European Search Report re EP app. No. 00938130, dated May 19, 2003.

"Hicor Nicellog V7.5," retrieved from Internet: http://www.siemens.ie/enterprise/apps_solns/pdf/Nicellog.pdf.

"CR-500 Series, Advanced Magneto Optical and DAT Communication Recorders," retrieved from the Internet: <ftp://ftp.teac.co.jp/pub/ipd/pdf/English/CR-500E.pdf>.

Supplementary partial European Search Report re EP app. no. 00938130.2-2414/US0015419, dated Jan. 7, 2003.

* cited by examiner



U.S. Patent

Mar. 22, 2005

Sheet 2 of 5

US 6,870,920 B2

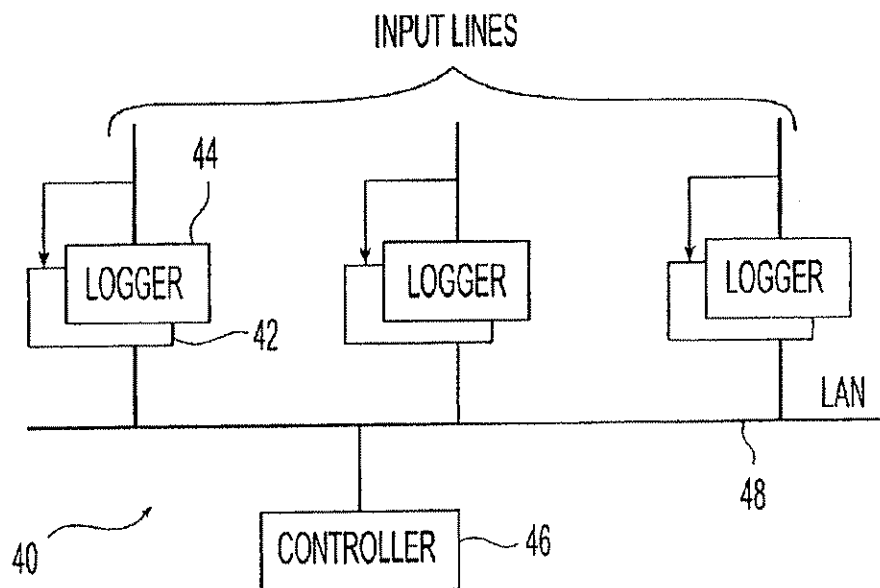


Fig. 2
(Prior Art)

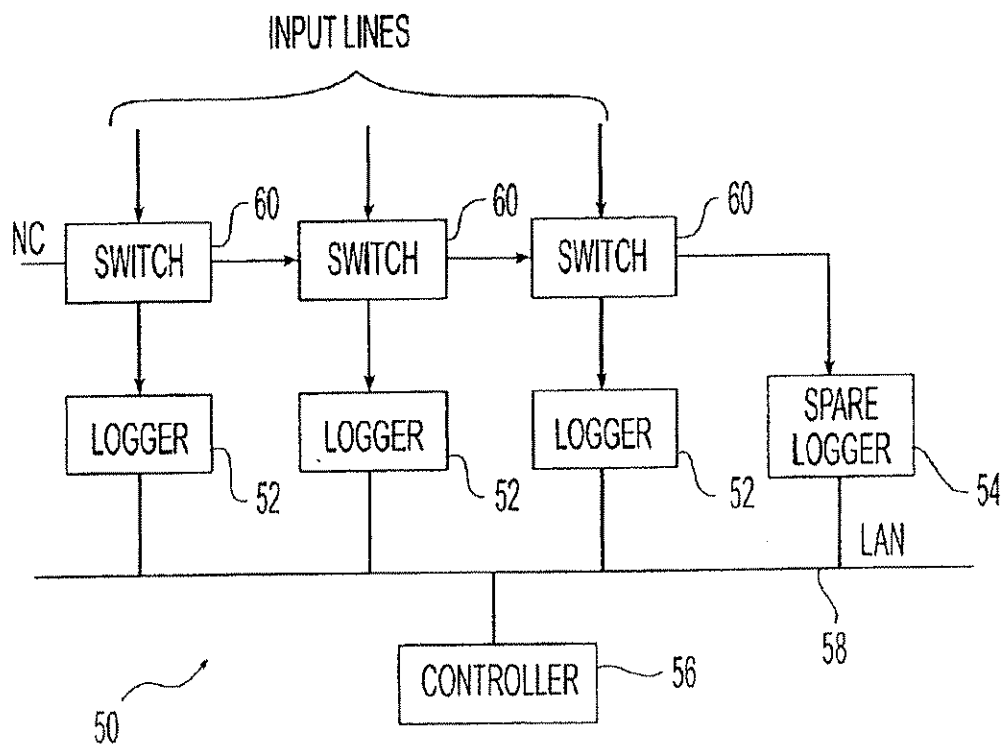


Fig. 3
(Prior Art)

U.S. Patent

Mar. 22, 2005

Sheet 3 of 5

US 6,870,920 B2

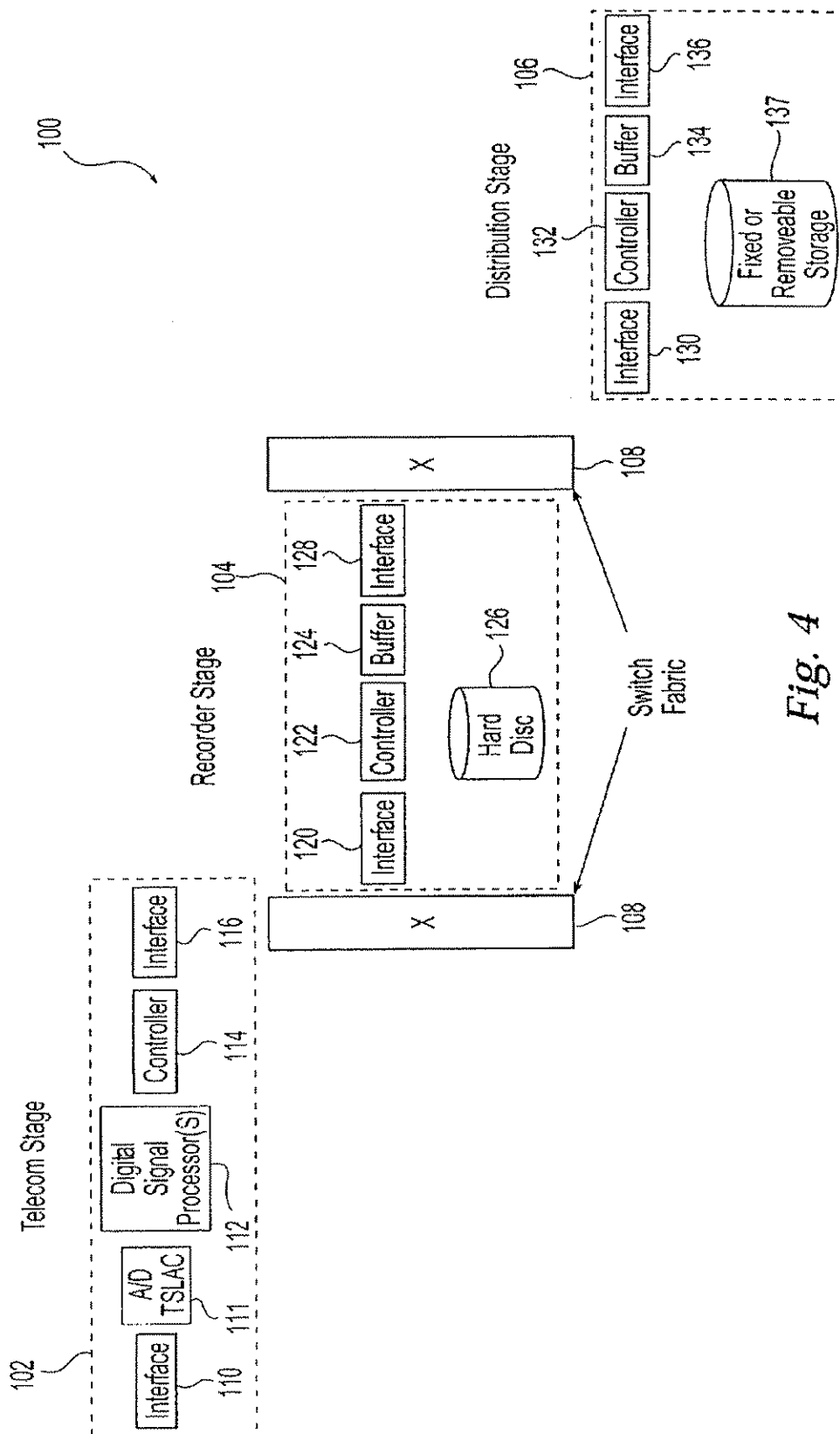


Fig. 4

U.S. Patent

Mar. 22, 2005

Sheet 4 of 5

US 6,870,920 B2

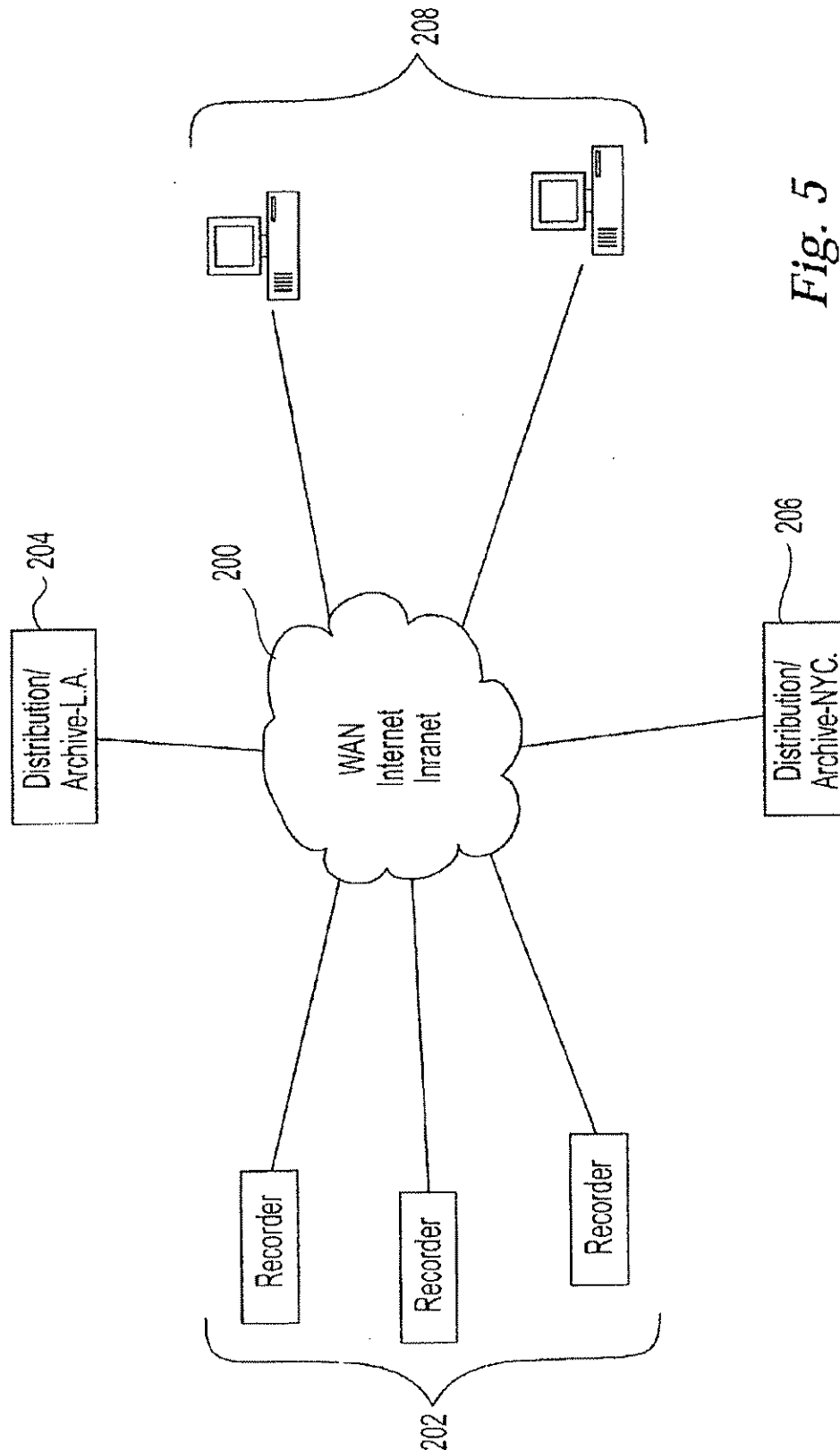


Fig. 5

U.S. Patent

Mar. 22, 2005

Sheet 5 of 5

US 6,870,920 B2

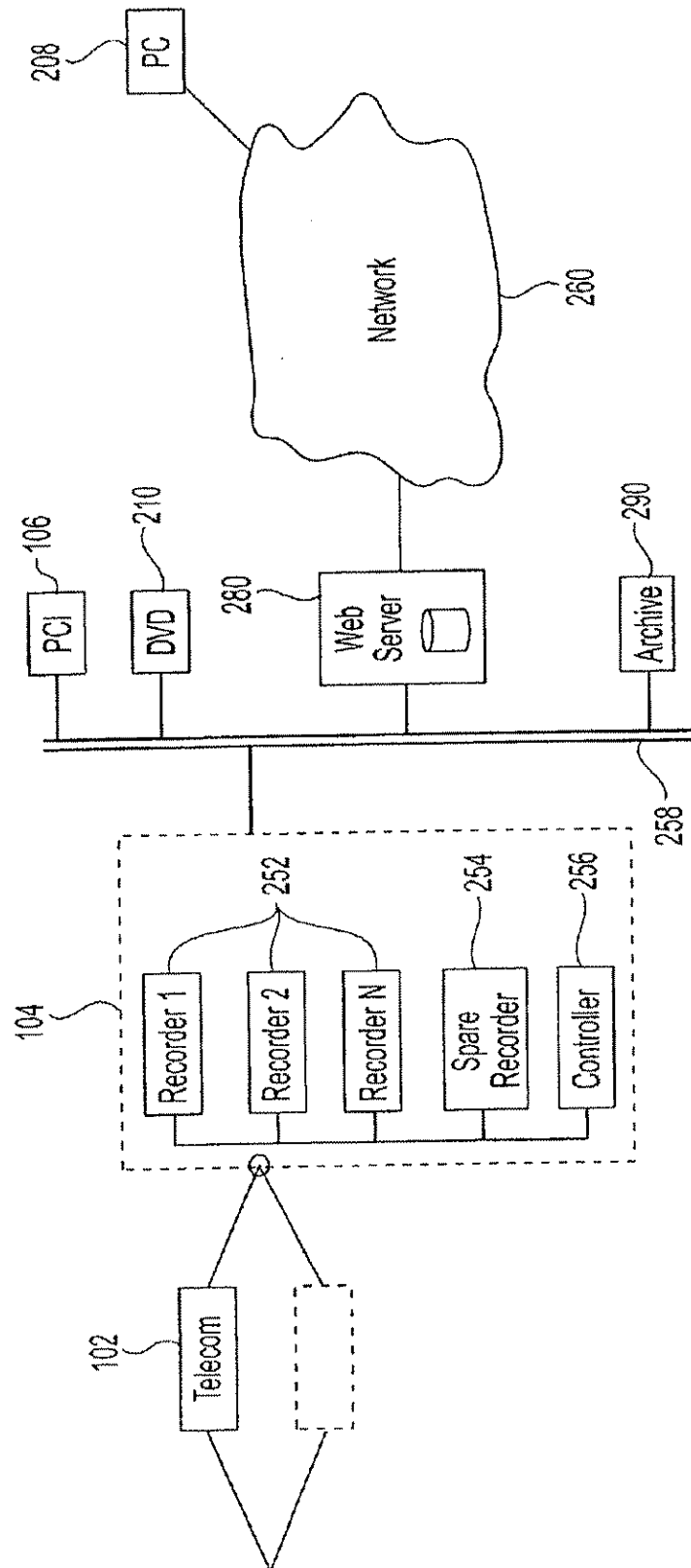


Fig. 6

US 6,870,920 B2

1

SYSTEM AND METHOD FOR MULTI-STAGE DATA LOGGING

This is a division of application Ser. No. 09/324,459, filed June 2, 1999.

FIELD OF THE INVENTION

The present invention relates generally to data logging, and more particularly to a flexible and cost-effective multi-stage system and method for processing multi-channel input data with improved fault tolerance and flexible user access.

BACKGROUND OF THE INVENTION

The efficient storage and retrieval of multi-channel data communications, and especially of voice, are critically important in many modern business and government applications. For example, financial institutions record instructions from clients as a protection against fraud and as evidence in legal proceedings about the content of telephone conversations; public safety agencies record emergency calls for event reconstruction and future investigations; commercial entities monitor transactions over the phone to evaluate salespersons' efficiency, to ensure customer satisfaction and to develop training programs. There is a growing need for reliable recording of multiple channels of multimedia information. These are but a few examples of applications in which it is necessary to efficiently store and process, usually at one location, multiple and frequently simultaneous communications from a large number of incoming data channels. Recent advances in multimedia applications further dictate the necessity to develop practical tools to efficiently process multiple channels in which incoming data can be in different formats, i.e., sound, images, data, etc. It is also apparent that with the proliferation of different communications media, computer platforms and operating systems, a very important practical aspect of all these applications is the ability to provide seamless and efficient interface between the multi-channel data system and its users.

Thus, while capturing the incoming information remains the main function of modern multi-channel data processing systems, other desirable system functions, such as the efficient storage, indexing and retrieval of recorded communications, are becoming increasingly important. Preferably, all system functions should be transparent to the users, regardless of the data format, of the communications media or the specific computer platforms being used. The present invention addresses the need for such a data logging system and method and illustrates their use in practical applications.

Data logging systems for capturing and recording massive volumes of data transmitted over multiple communication lines are known in the art. A typical prior art data logging system, such as the one shown in FIG. 1, is implemented as a single physical unit performing most or all required functions, including telephone interface, signal processing, random access buffering, and archive storage. In many practical applications such implementation is perfectly adequate. Examples of such systems are provided in, for example, U.S. Pat. Nos. 5,819,005; 5,448,420; 5,446,603; 5,396,371; 5,339,203 and 4,827,461 to the assignee of the present application. These patents are hereby incorporated by reference for all purposes.

However, as market needs change and performance demands increase, improved logger architectures have to be designed to support expanding capacity requirements and

2

emerging processing needs. Two strong market demands render prior art logger architectures at a disadvantage in this regard: the need to increase the number of input ports, and the need to provide simple and efficient access to the recorded information in platform- and media-independent ways.

Turning first to the input port requirements, modern data logging applications dictate the need to support an ever-increasing number of input channels. It should be apparent that as the number of channels increases so does the complexity of the processing system. The increased complexity in turn creates at least two potential problems: (a) diminished fault tolerance of the system; and (b) practical constraints on the physical design, including the weight and size of the system, its wiring and power requirements, among others.

Increasing the system's complexity generally results in a greater vulnerability and higher risk of data losses because of the increased probability that one or more system components can malfunction. However, people of skill in the art would recognize that in many applications it is critically important that the operation of the system remain error-free. Because of this stringent requirement and the fact that the information that loggers are entrusted to record is almost always original and ordinarily cannot be artificially regenerated, it is necessary to build into the system sufficient redundancy so that the malfunction of one or more components would not lead to a shutdown of the entire system. This in turn makes it increasingly important to devise a scalable system that provides fault tolerant characteristics regardless of the number of input channels.

Fault tolerant architectures that require additional "standby" hardware to replace failed components are known in the art. One such architecture, shown in FIG. 2, utilizes a local area network (LAN) 48 for communication between loggers 44 and controller 46. Each logger is responsible for recording communications on a plurality of input lines. In order to provide backup logging capabilities in the event a logger 44 fails, spare loggers 42 are provided. In particular, each logger 44 is associated with a spare logger 42 that is kept ready for use in the event the working logger 44 fails. This fault tolerant architecture is wasteful and impractical, because it requires more space and twice the hardware (at twice the cost) of a comparable standard logger system.

An improved architecture of a fault tolerant system, shown in FIG. 3, is described in an application by Yosef, EP 0822696A2, the content of which is hereby incorporated by reference. The Yosef approach is to use a single spare logger to back up a group of N loggers, thereby reducing the hardware cost in the case when $N \geq 2$ compared with the system shown in FIG. 2. Notably, however, even in this improved prior art architecture the spare unit is a completely functional device capable of performing all logging functions. Accordingly, it does not take advantage of the fact that in operation certain components of the logger fail much more frequently than others. This is the case, for example, in components with physically moving parts, such as hard drives. In accordance with the present invention cost savings are realized for the same fault tolerance level using separation of the logger into two or more functional stages, where higher redundancy is provided for the components having higher failure rates. As illustrated below, using this approach it is possible to reduce the cost of the system considerably, without affecting its performance level.

As mentioned above, the increased complexity of the system also poses wiring, size and other practical problems.

US 6,870,920 B2

3

For example, an important consideration when installing a logger with a high number of input channels is the number of wires that interconnect the logger to the PBX. Thus, a logger having 128 input channels requires at least 256 wires. For practical reasons including susceptibility to picking up audible electrical noise over long cable runs, it is desirable to install the logger next to or near the PBX. However, this location may not always be optimal, desirable or even possible in practice. In particular, spatial constraints as well as security issues pose serious problems. Thus, the large size of a high-capacity logger can make it impossible to install it in certain locations. It may further be costly and/or impractical to run massive cable bundles from the PBX room to the loggers. For security reasons it may not even be desirable that the logger is anywhere near the PBX. It should thus be apparent that prior art systems implemented as single physical units suffer from some very serious practical problems. In accordance with the present invention, separation of the logger into several functional stages provides the flexibility to perform different functions at spatially different locations, i.e., in different parts of the same building, city, or the world. In view of the above, the potential advantages of this approach should readily be apparent.

In addition to the requirements for increased capacity, a whole new set of problems is created by the market demand for flexible playback or data access to the stored information. In the context of this application, access to the recorded data is referred to as data "distribution". People of skill in the art will appreciate that it is by no means a trivial task to design a system capable of recording and processing multiple channels of information, while at the same time providing concurrent access to the recorded information from a large number of users. The task becomes even more complex if the users employ different computer platforms.

While many attempts have been made in the past to address these and other problems associated with prior art data logging, no adequate solution has been proposed so far. The system and method of this invention overcome problems associated with the prior art and thus are believed to present significant technological advance. In particular, separation of the logger into several functional stages provides considerable flexibility in designing a functionally and ergonomically optimized system for use in various practical applications. Further, distributing logger functions into separate physical stages leads to significant and unobvious advantages over the existing practice of utilizing a single physical unit. For example, hardware for performing certain functions in the same system can be bulky and thus inconvenient for positioning at a particular location. Due to the functional separation approach of the present invention, however, bulky parts can advantageously be placed where appropriate. Thus, the relatively bulky recording equipment capturing transaction information in the New York Stock Exchange can be placed miles away from the cramped floor of the Exchange. Additionally, in accordance with the principles of the present invention the design of the distribution stage enables users to access the recorded data at their convenience either real-time or at a later point. Importantly, fault tolerance can be provided at a fraction of the cost associated with prior art approaches. The proposed approach provides better system scalability, reliable performance at a low cost and great flexibility in the retrieval of stored information compared with existing designs.

SUMMARY OF THE INVENTION

Accordingly, it is an object of the present invention to provide a multi-stage data logging system that overcomes

4

problems associated with prior art solutions, and helps meet the market demands for increased input capacity, high distribution flexibility and fault tolerance.

In a preferred embodiment, the invention is a multi-stage data logging system comprising: a telecom stage receiving input from a plurality of input channels; a recorder stage having one or more recorders, at least one recorder storing data associated with input received from at least one of said plurality of input channels; a distribution stage providing access to data stored in the recorder stage; a first interface linking the telecom and the recorder stages and a second interface linking the recorder and the distribution stages; wherein at least two stages of the system are physically separable and in operation can be located wide distances apart.

In a specific embodiment the telecom stage of the system further comprises a first interface capturing signals from said plurality of input channels; one or more signal processors converting captured signals into data having a predetermined format; and a second interface for transmitting said converted data to said recorder stage. In embodiments where the input signal can be analog, the telecom stage of the data logging system further comprises at least one analog to digital signal converter.

In specific embodiments, the data logging system of the present invention provides data compression; time stamping of the received input; authentication of signals from the input channels; and encryption of the converted data.

In a preferred embodiment the data logging system of the present invention comprises an archive storage device, part of the recorder and/or telecom stages, for archiving data. In specific embodiments this device can be fixed, such as a RAID array, or removable.

In another preferred embodiment, the recorder stage of the data logging system comprises at least one backup recorder, and the system has means for detecting malfunctions in recorders of the recorder stage, and for automatically switching interface links from the detected malfunctioning recorder to the backup recorders. Related to this embodiment is another aspect of this invention, which is a method for operating a multi-stage data logging system comprising: detecting a malfunctioning recorder in the recorder stage; automatically switching interface links from the detected malfunctioning recorder to a backup recorder to ensure uninterrupted operation of the system; and without disrupting the operation of the system replacing the detected malfunctioning recorder with a functioning recorder.

In yet another aspect, the invention is a method for increasing the capacity of a multi-stage logger system comprising: without disrupting the operation of the system attaching to a network-based or four-wire-based interface between the telecom and the recorder stages at least one recorder so that the combined capacity of the recorders in the recorder stage is equal to or exceeds a given number of channels; and/or attaching at least one additional telecom block to increase the input channel capacity of the system.

In another preferred embodiment, the present invention is a data logger, comprising: a telecommunication device receiving input from a plurality of data sources; a processor converting input from said plurality of data sources to one or more data formats; a memory for storing converted data corresponding to the received input from said plurality of data sources; a communication path; and a server transferring stored data from one or more of said plurality of data sources via the communication path to at least one remote user. In specific embodiments the server is a Web server and the communication path is the Internet.

US 6,870,920 B2

5

In another aspect, the present invention is a method for accessing information in at least one digital logger storing data associated with input from a plurality of input channels, comprising: at a Web server having access to said at least one digital logger, receiving a request for retrieval of stored data from a client; retrieving stored data in accordance with the received request; and transferring the retrieved data to the client. In the specific embodiment covering voice input channels, the method further comprises accessing a record of an input channel made by a digital logger; or accessing call information for a record of an input channel made by a digital logger.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be understood and appreciated more fully from the following detailed description, taken in conjunction with the drawings in which:

FIG. 1 is a schematic diagram of a data logger of the prior art incorporating multiple logging functions in a single unit.

FIG. 2 is a schematic diagram of a logging system of the prior art, employing full operating redundancy where for each working logger there is a spare logger;

FIG. 3 is a schematic diagram of a logging system of a prior art employing one spare logger for each group of "N" working loggers.

FIG. 4 is a schematic diagram of a multi-stage data logging system in accordance with a preferred embodiment of the present invention.

FIG. 5 is a block diagram of a multi-stage logging system configured in accordance with a preferred embodiment of the present invention.

FIG. 6 is a schematic diagram illustrating another preferred embodiment of the multi-stage logger in accordance with the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Prior Art

FIG. 1 is a schematic diagram of a prior art data logger 10. Logger 10 incorporates an interface 14 for receiving data from input lines 12, an analog/digital (A/D) and a digital/analog (D/A) signal converter 16 that converts analog signals from input lines 12 into digital signals when data is flowing in one direction and digital to analog when the signal is flowing in the opposite direction. Interface 14 also communicates with a speaker 17. The system further comprises a digital signal processor 18, generally used to compress the digital signals from converter 16; a controller 20, a buffer 22, and hard disc 23. In a typical implementation all components are located in a single physical unit 13. As shown in FIG. 1, the unit may also include a storage unit 24, such as a digital audio tape (DAT). The configuration and operation of a logger of the type illustrated in FIG. 1 is described in detail in several U.S. patents, including U.S. Pat. Nos. 5,819,005; 5,448,420; 5,446,603; 5,396,371; 5,339,203 and 4,827,461 to the assignee of the present application, and which are incorporated herein by reference.

It will be appreciated by people of ordinary skill in the art that the number of input channels that can be handled by a logger of the type shown in FIG. 1 is usually fixed by the design constraints of the physical unit, i.e., by the number and the capacity of the interface cards, the configuration of the processor 18, and others. Accordingly, to process a higher number of input channels it is generally necessary to

6

either increase the number of interface cards (limited by the number of available slots) or duplicate the logger 10 one or more times, such that each unit handles a particular group of incoming channels. In the configuration shown in FIG. 1, as well as multi-logger architectures in which logger 10 is replicated to increase input channel capacity, users may encounter various problems of the type discussed in the background part of this application. For example, failure of one block in logger 10 may disable the entire unit and thus jeopardizes the recording of several input channels. Thus, the fault tolerance of the system depends on its weakest link, i.e., on the most failure-prone component of the unit. In many cases where no backup unit is provided the resulting fault tolerance is unacceptable. Further, space constraints may pose problems because the standard design of the logger 10 as one physical unit does not allow its separation into functional blocks that can be placed in spatially optimal locations.

Reference is now made to FIG. 2, which is a schematic diagram of a prior art fault tolerant data logging system. Data logging system 40 illustrated in FIG. 2 employs one spare logger 42 for each working logger 44. In a particular embodiment, each logger can be implemented as illustrated in FIG. 1 but other embodiments are possible, as known in the art. In this architecture, all loggers are connected via a local area network (LAN) 48 and upon failure of a working logger 44, the spare logger 42 assigned to the failed logger is automatically activated. While just about any fault tolerant system will require some additional "standby" hardware to replace failed components, system 40 illustrated in FIG. 2 requires twice the hardware, at twice the cost, of a conventional system and in most cases is commercially impractical.

Reference is next made to FIG. 3, which is a schematic diagram of an improved fault tolerant data logging system of the prior art. System 50 illustrated in FIG. 3 consists of one or more groups of "N" loggers 52 connected via LAN 58, and linked by a group of "N" switches 60 to a single spare logger 54. When one logger 52 fails, the switches 60 automatically reroute data from the failed logger's input lines to the spare logger 54. For $N \geq 2$ this system requires less "standby" hardware than the prior art system of FIG. 2. Nevertheless, it is apparent that such a design is suboptimal, at least because it does not make use of the fact that certain components of the logger are more reliable than others. Importantly, this and other prior art logger designs represent a single-unit design philosophy that carries many performance limitations as noted above.

The Present Invention

In contrast to the prior art designs described above, the approach taken in accordance with the present invention is to combine low-cost and high-reliability multi-stage recorders storing data from multiple input data channels with a powerful distribution technology that enables platform-independent access to the stored data from different physical locations. Multiple channel data recording, and particularly voice loggers, are known in the art. As shown above, some steps have already been taken to improve the reliability of such devices using redundancy approaches. From the distribution perspective in the last few years Internet technology is becoming standard. However, to the best of this inventor's knowledge, there is no suggestion in the past of combining these two paradigms in a multi-stage distributed logger, a next-generation product that brings together the most desirable features of these technologies.

In particular, loggers have been used in the past to record, index and store large volumes of data from different input

US 6,870,920 B2

7

channels. However, practical limitations, such as their single-unit design, have limited the use of such loggers in many applications. With the distributed multi-stage architecture of the present invention, at least the following advantages over the prior art can be expected:

(a) highly reliable and low-cost data recording and storage due to a novel fault tolerant architecture in which failure-prone components are duplicated for redundancy;

(b) optimized use of space made possible by the physical separation of the logger in separate stages that allow, for example, placing bulky components away from locations where their presence can be inconvenient (due to space limitations), or even highly undesirable (as in covert operations);

(c) improved business models for information distribution.

Reference is now made to FIG. 4, which is a schematic diagram of a multi-stage data logging system in accordance with a specific embodiment of the present invention. In the illustrated embodiment, system 100 consists of a telecommunications stage ("telecom stage") 102, recorder stage 104, and distribution stage 106. Broadly, the telecom stage serves to capture incoming signals, convert the captured signals in a predefined format, preferably digital, and perform certain processing on the converted signals, usually in the nature of compression. Recorder stage 104 stores the processed input signals to a hard drive or similar large-capacity storage device, as well as in a buffer from which the data can be forwarded to fixed or removable storage devices, which in the embodiment shown in FIG. 4 are part of the distribution stage. In accordance with the present invention, the distribution stage serves not only for access to the stored information but also for archiving purposes. Importantly, individual functional stages of the system 100, such as those illustrated in FIG. 4, can be physically separated and typically are housed in separate physical units.

It should be understood that FIG. 4 is merely an illustration of a system built in accordance with the principles of the present invention and should not be construed as a limitation of its design approach. Thus, functional separation of the logger into more than three stages can be desirable in certain applications. In alternate embodiments functional stages can be combined—for example, with reference to FIG. 4, the telecom and recorder stages on one hand, or the recorder and distribution stages on the other hand can be implemented as single physical units. Furthermore, in alternative preferred embodiments data from the recorder stage is made available to users via a communications network such as private communication lines, corporate intranets, the Internet or any combination thereof. It will be appreciated that such networks can be accessed by various different computers, so that stage 106 is "distributed" and would have no identifiable physical location.

In another aspect of this invention the stages of the system shown in FIG. 4 are connected using switch fabric 108 interfacing the telecom and recorder stages on one hand, and the recorder stage and distribution stages of the system 100 on the other. It will be appreciated that switch fabric 108 interfacing different stages can be different in design and implementation. In accordance with the present invention, the selection and design of the switch fabric 108 linking the individual stages of the overall system 100 are design parameters that allow for the physical separation of the individual stages in accordance with the preferred embodiments. In particular, dependent on the application, the stages of system 100 can be physically separated so that, for

8

example, the telecom and recorder stages can be located in different parts of the same room, in different rooms of the same building, in different buildings of the same town, in different towns or even different continents. In a specific embodiment such wide spatial separation can be implemented using, for example, standard public switched network access arrangements, dedicated lines, or other communications media, as known in the art. The individual stages of the system in accordance with the present invention are considered in more detail next.

The Telecom Stage

In accordance with the present invention telecom stage 102 generally functions to capture and pre-process signals from a plurality of communications lines into a format that is recognized by the recorder stage. As indicated above, in a preferred embodiment the communication lines could be standard telephone lines, dedicated communication lines or other input data sources, which may transmit analog or digital signals. Regardless of their physical source or data format, in a preferred embodiment the captured input signals are digitized and compressed, as known in the art. Naturally, different types of input data can be transformed into different internal formats, as instructed by the user.

More specifically, as shown in FIG. 4, telecom stage 102 incorporates a first interface 110 for receiving signals from the input lines (not shown), which typically are telephone lines. In accordance with a preferred embodiment, interface 110 is designed for analog or digital input signals, or some combination thereof. When used with telephone lines, interface 110 generally is a high impedance interface that allows for passive tapping of the phone lines. In this embodiment interface 110 employs voltage and/or current sensing to trigger recording of corresponding input channels. External triggering signals can also be used at the telecom stage, as known in the art. In the case of a digital input, interface 110 serves as a digital phone coupler, as known in the art. Various alternate embodiments of the interface 110 are known in the art and are described in detail, for example, in U.S. Pat. No. 4,827,461 to Sander, which is incorporated herein by reference. It will be appreciated that interface 110 can be used for various additional functions, such as beep generation, notching and others.

In the embodiment illustrated in FIG. 4, telecom stage 102 further comprises an A/D signal converter 111 for digitizing the received analog signals. Naturally, no such converter is required for purely digital inputs. Stage 102 also comprises a digital signal processing (DSP) unit 112 for processing signals digitized by converter 111 or supplied directly from digital input channels. In a preferred embodiment, DSP unit 112 is capable of handling various input data types, i.e., voice, image, facsimile and others. In a specific embodiment one processor 112 is capable of handling various input data types and its processing algorithm is selected dependent on the input data type. In alternative embodiments, one processor can be used for each input channel so that different processing algorithms can be applied simultaneously to different input channels. It will be appreciated that whether one or more processors are used, different data compression algorithms can be applied to different input channels dependent on the data type, the statistical distribution of the incoming data, and other factors. Such processing algorithms are generally known in the art and will not be considered in further detail.

Telecom stage 102 illustrated in FIG. 4 further comprises controller 114 that operates to direct and monitor the func-

US 6,870,920 B2

9

tions of the entire stage. Specifically, in a preferred embodiment, controller 114 directs the application of a particular processing algorithm by DSP processor 112 to a particular input data type. When instructed by the controller, in a specific embodiment processor 112 is capable of performing DTMF signal detection and decoding, and preferably also performs compression of the data in accordance with well established standards. For example, in the case of voice signal recording, the output of DSP processor 112 is preferably made compatible with the G 723.1 and G 722 voice compression standards, which are known in the art.

In a specific embodiment, interface 110 and A/D converter 111 together with a time slot assignment circuit (not shown) are implemented as a logger coupler card, that places the input data on a pulse code modulation (PCM) highway (not shown) within the telecom stage 102. In this case the DSP unit 112 is directed by the controller 114 to process signals in the appropriate time slots. In particular, controller 114 reads the system clock, directs the assignment of time slots to individual input channels, and provides a time stamp that is used in the creation of individual channel records. As noted above, block 114 also controls the operation of the DSP unit 112 and distributes data compressed at its output to interface 116. In a preferred embodiment an important function of the controller is to reconfigure the processing algorithm(s) of the DSP unit 112 using downloaded software. Thus, in a specific embodiment an input port (not shown) is provided for the controller to download instructions used to re-program the DSP processor(s) 112 from a personal computer (PC).

Finally, in a preferred embodiment telecom stage 102 comprises a second interface 116 for transmitting compressed data to the recorder stage 104. In a specific embodiment the second interface 116 may be an E1 (2 Megabit) transceiver or, for smaller applications, an RS-485 (1 Megabit) line. Other embodiments may be used, as known by those of skill in the art.

In accordance with the present invention, certain functions that are not required but are frequently useful in the operation of data loggers are also provided in specific embodiments. Such functions include, for example, time stamping, encryption and authentication of incoming data. Generally, authentication refers to mechanisms by which the transacting parties prove they are who they claim to be, i.e., in this context that data from an input channel indeed comes from a particular source; encryption usually refers to the altering of data so that it cannot easily be read or modified, if intercepted. If used, such functions should be placed as close as possible to the source of the information that is being captured—and accordingly are implemented in the system of this invention as part of the telecom stage 102. In a specific embodiment, these functions are implemented by the controller block 114. Other functions of the telecom stage can be implemented in specific embodiments, as known in the art. Thus, for example, in a specific embodiment telecom stage 102 can provide lightning protection and can be used to detect DTMF signals, to identify caller IDs, to detect the presence or absence of voice in a particular input channel, and correspondingly control a trigger mechanism for use by the recorder stage 104. As known in the art, the telecom stage can also be used as a current and/or voltage sensor. Functions described in this paragraph can be implemented as part of the logger coupler card used in a specific embodiment, and/or the DSP unit 112 of the telecom stage 102.

It will be appreciated by people of ordinary skill in the art that in general the components of the telecom stage are

10

robust and in particular involve no physically moving parts. Accordingly, in a preferred embodiment, they need not be duplicated in fault-tolerant architectures built using the principles of the present invention. Naturally, in very sensitive applications where there is little or no margin of error the telecom stage can also be duplicated with a standby unit, as shown in FIG. 6. However, in accordance with the present invention, a single standby telecom stage can be used to serve as backup of N recorder stages in a manner similar to that illustrated in FIGS. 3 and 6. The cost savings compared with the architecture shown in FIGS. 2 and 3 are readily apparent.

Finally, since telecom stage 102 is separated from the remaining components of the logger by switch fabric 108, the stage can be implemented in a separate physical unit having as many input channel slots as desired. This compares favorably to the standard single-unit logger design, in which the number of input channels is limited by the number of available slots on the circuit board, which in turn are limited by the overall dimensions of the unit. It will be appreciated that the multi-stage design used in accordance with the present invention thus extends the input channel capacity of the logger.

The Recorder Stage

Again with reference to FIG. 4, recorder stage 104 of the logger stores input from the telecom stage to a hard drive or a suitable large-capacity random access storage device, as well as in a buffer from which the data can be forwarded to fixed or removable storage devices. In a particular embodiment directed to storing voice records, the function of recorder stage 104 can be described broadly as creating voice files and providing an associated database with stored call information. To this end, the recorder stage 104 in a preferred embodiment has the ability to create new voice records, merge call records and establish a database with call information for each created record. In a preferred embodiment, newly created records, along with identifier information are stored in a database at unique record addresses.

In accordance with the embodiment illustrated in FIG. 4, recorder stage 104 has the following components: (a) an interface 120 receiving input from the telecom stage; (b) a controller 122 directing and monitoring the recorder stage operations, and generally comprising a microprocessor with memory; (c) an elastic buffer 124 for transitional data storage; (d) a hard disk or similar large-capacity storage device 126 for data storage (2 Gigabytes minimum); and (e) a second interface 128 for connection to the distribution stage. It will be appreciated that while the components of the recorder stage are illustrated in FIG. 4 as being separate, they need not be implemented in separate physical units.

The function and operation of the recorder stage 104 is generally known in the art. For example, it is described in U.S. Pat. Nos. 5,819,005; 5,448,420; 5,446,603; 5,396,371; 5,339,203 and 4,827,461 to the assignee of the present application, which are hereby incorporated by reference for all purposes. It will be appreciated that alternate embodiments employing different storage media or internal architecture can be used, if necessary. For the purposes of this invention, it is important to note two main differences from prior art designs. The first difference is due to the flexible platform-independent data access provided in accordance with the present invention to end users in the distribution stage. To support such an access, when viewed from the distribution stage the recorder of this invention appears as a

US 6,870,920 B2

11

file server providing access to a plurality of records (from input data sources) stored in a database at unique record addresses. The server preferably supports different data transfer protocols. The second difference is that since the recorder stage uses moving physical parts and is thus prone to malfunctioning, for high reliability in accordance with this invention it is desirable to provide backup units of the N recorders (or critical components thereof of) the recorder stage. These two aspects of the present invention are described in further detail next.

In accordance with a preferred embodiment, recorder stage 104 has operating system that supports different file access protocols, such as Microsoft's SMB, the UNIX-based NFS and the standard FTP file transfer protocols. Thus, when viewed from the distribution stage 106, the recorder in a preferred embodiment appears as a server supporting these and possibly other file transfer protocols, in which records that correspond to different input channels appear to the user as separate files having unique record addresses. In a preferred embodiment, these files are identified by the call record information and contain, for example, information about the caller ID, the date and time of the communication, its duration, and others. With reference to FIG. 6, a user at a PC 106 connected to the recorder stage via communications path 258 and operating, for example, in a Microsoft Windows environment can click on a "Network Neighborhood" icon on his computer to locate a recorder 202 of interest. As noted above, various records in such a recorder would appear as files that are viewable using Microsoft's SMB protocol. Once a desired record is identified, it can be "dragged" to the user's computer for playback. Similarly, a NFS or Linux client can access files at a recorder that supports the corresponding UNIX-based protocol. Further, file transfers from the recorder can be accomplished over a network 260, such as the Internet, using the well-known FTP protocol. Naturally, security can be implemented at the recorder end, if desired, as known in the art. In each case, it is important that from the viewpoint of the user the recorder stage appears as a server accessing different files organized in such a manner as to provide the user with access to unique records from different input channels.

Focusing next on the fault-tolerant aspect of this invention, FIG. 4 illustrates a recorder stage with a single recorder. As noted above, however, in many practical applications the recorder stage can have two or more recorders, each processing information about a particular set of input channels. Furthermore, in terms of the fault-tolerant aspect of the present invention and with reference to FIGS. 2, 3 and 6, it is noted that unlike the system's telecom stage, the recorder stage typically comprises physically moving parts, and thus is prone to occasional malfunction. Accordingly, in a specific embodiment of the present invention shown in a diagram form in FIG. 6, the recorder stage is implemented in a single physical unit in what is known as a "hot swapping" MxN design. In this design one backup recorder unit 254 is used for every N working units 252, and switchover to the backup unit occurs automatically without turning off the power. In a specific embodiment, N recorders implemented as swappable modules are physically placed in drive bays of a physical unit and can be replaced, if necessary, without turning off the power of the unit. Detection of a recorder malfunction and control over the entire stage is provided by the controller 256, as known in the art. Schematically, the architecture of the fault-tolerant embodiment of the present invention resembles the architecture in FIG. 3, the important difference being that instead of using an entire backup logger, the system of the present invention

12

uses as duplicate hardware merely the recorder stage or components thereof, such as the hard disk 106. Such a design approach not only provides the desired performance level and fault tolerance but also results in considerable savings in terms of the overall system cost.

Providing automatic switchover without disrupting the operation of the system, as described, is an important aspect of the present invention. This feature is made possible by the selection, in a preferred embodiment, of network or 4-wire interfaces linking individual stages of the logger system, which enables inserting or taking out system components without affecting the operation of the entire system. It will be appreciated that the same feature would be difficult to provide with PC bus interfaces. Further, it will be appreciated that this selection of the interface links also makes it possible to incrementally add recording capacity simply by attaching to these links of additional telecom blocks, recorders or both. Clearly, this incremental addition can also be done without interrupting the operation of the system, which feature of the present invention is perceived to have significant practical utility.

The Distribution Stage and Switch Fabrics

In accordance with the present invention a very flexible approach is taken to the design of the distribution stage of the logger. Generally, the distribution stage serves for retrieval of recorded information and providing it in a humanly recognizable form, i.e., as an image, a printout, a sound clip or others. In a preferred embodiment, the distribution stage also serves for archiving the recorded information to a removable storage, such as magnetic tape, magneto-optical storage device, DVD, or others.

As best seen in FIG. 6, in a preferred embodiment the distribution stage may have no single physical location, and in this sense is truly "distributed". As seen in this figure, one or more recorders 252 are attached to a communication path 258. This path can be provided using Ethernet, optical fiber or other media. It may be part of a local area network (LAN), a wide area network (WAN), and preferably has access to the Internet. In a specific embodiment, the communication path is implemented as a Universal Serial Bus (USB). It is foreseen that the communication path may also be part of a storage architected network (SAN). As seen from the above list of options, there is no limitation on the type of communication path provided at the back end of the recorder stage, which path concurrently serves as part of switch fabric 108 (see FIG. 4).

With reference to the specific embodiment illustrated in FIG. 4, the distribution stage also has a controller 132 for directing and monitoring distribution stage operations, a buffer 134 for transitional data storage, and a second interface 136 for distributing data to one or more output channels. Distribution stage 106 in the embodiment illustrated in FIG. 4 may also include an archive storage device 137 for archiving data, which can be either fixed (such as a RAID array) or removable. In one embodiment, this archive storage device 137 is a DVD RAM drive, a digital audio tape (DAT), or any suitable device capable of storing large volumes of data from multiple recorders of the recorder stage 104. In an alternate embodiment, the archive drive is part of the recorder stage, not the distribution stage. As seen, in this "OEM-friendly" embodiment, users may be free to design their own customized distribution stage.

With reference to FIG. 6, in a preferred embodiment, the distribution stage is implemented using at least in part Web server design technology. This approach has the advantage

US 6,870,920 B2

13

of using a powerful distribution tool, which is well known and can readily be employed to support a number of users regardless of their specific computer platforms. In this sense, the Web server 280 acts as an intermediary between one or more recorders 252 in the recorder stage of the logger, and the users accessing the stored information via, for example, the Internet. As noted in the preceding section, in accordance with the present invention the recorder stage will make records from different input channels available as files that are stored at unique record addresses and are made accessible to the Web server 280. In this context it should be noted that in a preferred embodiment the Web server used in the present invention has built-in archiving functions.

In accordance with the present invention Web server 280 may be implemented using any computer, such as, for example, a SUN work station using the UNIX operating system and running a web server program that preferably accepts requests for information framed according to a suitable protocol, such as the HyperText Transport Protocol (HTTP) or a version of it that supports public-key-based authentication or encryption. In response to these requests, Web server 280 accesses the loggers records directly, or it causes a process to access data in a database of the recorder stage through a common gateway interface (CGI); and sends the requested files to the requesting client according to the client's Internet address which, in one embodiment, may be provided according to the Transmission Control Protocol, Internet Protocol (TCP/IP). In a specific embodiment, access to the server can be provided after going through a firewall (not shown) for added security. It will be appreciated that similar distribution scheme can be provided as part of a corporate intranet.

In a specific embodiment employing a Web server as part of the distribution stage users 208 access the Web server through a browser or another suitable application that can playback audio files, in the case of recorded voice information, or display data in a generic case. Specifics of the file formats or data transmission protocols are generally known in the art and need not be described in further detail.

Reference is now made to FIG. 5, which is a block diagram of a multi-stage logging system configured in accordance with a preferred embodiment of the present invention. A wide area network (WAN) 200, such as the Internet or alternatively an Intranet, links a series of recorders 202 located in various different cities, to distribution/archiving engines 204 and 206, located in principal cities like Los Angeles and New York, respectively. A series of user computers 208, possibly located in different cities, is also connected to WAN 200 and can access data recorded by recorders 202 and stored in archive drives 204 and 206. In this configuration, agents located in offices, or even their own homes, in different cities throughout the world can access data logged by equipment physically located thousands of miles away.

As noted above, with reference to FIG. 4 each stage of the logger in accordance with the present invention is designed so that the switch fabric units 108 can be used not only to provide physical separation of the individual stages of the logger, but also to achieve fault tolerance of the system by isolating failed components and switching operation to properly functioning standby units. Numerous configurations are possible, as illustrated in exemplary embodiments above.

Low cost components and switch fabric design also allow for fault tolerant capability in a scalable fashion. To demonstrate the fault tolerant characteristics of a typical

14

configuration, consider a recorder stage comprised of eight working recorders and three "spare" recorders. Assuming a 2 Megabit (4 wire) interface linking the recorder stage with the telecom and distribution stages, the eight recorders would be linked to the three spares via a 32x12 switch matrix ((8 working recordersx4 wires)x(3 spare recordersx4 wires)). In the event of a recorder failure, the recorder controller would route the failed recorder's input to a spare recorder via the switch matrix. The cost savings compared with the case of duplicating entire units are readily apparent.

The advantages of the data loggers built in accordance with the present invention are illustrated in the following examples:

EXAMPLE 1

One of the advantages of the multi-stage distributed logger of the present invention—its design flexibility—is clearly illustrated by its use in stock or commodities exchanges. Financial institutions represented at such exchanges record instructions from clients as a protection against fraud and as evidence in legal proceedings about the content of telephone conversations. Therefore, it is highly desirable that data loggers be provided to record communications between brokers at the exchange floors and their clients. As known, however, there is always a concern about the space on the trading floors. This concern is specifically addressed by the multi-stage design of the logger in accordance with the present invention, in which only the telecom stage electronics would need to be located physically on the trading floor. The recorder stage and various distribution servers of the logger could be located physically either someplace more convenient in the building, or in fact someplace else in the city or the world. For example, an E1 twisted pair interface can stretch approximately 2000 feet, so with available "repeaters" the separation distance between the telecom and the remaining stages is virtually unlimited. Since the voice compression function is performed in a preferred embodiment in the telecom stage, use of the public switched network (or private) facilities would be optimized. Importantly, the telecom stage can apply encryption algorithms so that data coming out of the exchange floor is protected against unscrupulous use. It will be appreciated that the same advantage can be used, for example, by law enforcement agencies that are entitled to have wiretap access to phone lines, but until now did not have a mechanism to protect themselves by encryption from being wiretapped themselves.

Furthermore, it can be appreciated that the multi-stage design of the logger in accordance with the present invention enables considerably simplified disaster recovery procedures, since destruction or simply failure of any single component would not affect the function of the entire system. In addition, since the present invention separates the telecommunications electronics from the recorder, cabling to the recorder can easily be reduced. Such cabling reductions may range from 15:1 to 200:1. The net result is lower installation costs and the satisfaction of any political concerns regarding system location and security.

EXAMPLE 2

In this example it is shown how the multi-stage design of the data logger in accordance with the present invention can be used to create and implement new business models. Assume for example that a data logger is used by a police to monitor 911 calls in a particular area. Information about such calls is generally available to the interested public, for

US 6,870,920 B2

15

example the news media, but so far has been used in only a few cases primarily because of the difficulties associated with the access to such information. Using the distributed design of a logger in accordance with the present invention, however, it would be a simple matter to make such records available to the public immediately, possibly for a fee.

In particular, a Web server run by the police can store or be given access to data files corresponding to individual 911 calls. With reference to FIG. 6, an authorized user may utilize a personal computer to access the police logger over the world wide web (WWW) of the Internet at a predetermined URL. The user's computer may be running standard web browser, such as the NETSCAPE browser. As soon as the computer is connected to Web server 280 a computer process starts communicating with the user through the web browser. In particular, a service routine may be initiated causing a "home page" to be displayed, which greets the user, and describes the service provided by system. Next, Web server 280 elicits user information, including a user identification (ID), password and other administrative data necessary for ensuring that the user is an authorized user. Next, the audio file(s) corresponding to individual 911 call records can be displayed to the user who can then select a file. As known in the art, various protocols exist for the playback of audio files over the Internet. By means of an example, the user can be provided with a File Header information defining the audio files in fields such as: (1) File name; (2) Date created; (3) Size in bytes; (4) Audio file types in use; (5) Total associated files; (6) a command set code; and others. Additionally, instructions may be provided for decompressing and decoding a specific or proprietary Audio Player software (possibly residing on the Web server), on how to play the files, including hyperlinks, etc. Various levels of access protection can be implemented, as known in the art, so that access hierarchy can be established for different groups of users.

It should be apparent that the use of the logger along the lines described in Example 2 creates the possibility of a completely new and heretofore unused business model.

While the foregoing has described and illustrated aspects of various embodiments of the present invention, those skilled in the art will recognize that alternative components and techniques, and/or combinations and permutations of the described components and techniques, can be substituted for, or added to, the embodiments described herein. It is intended, therefore, that the present invention not be defined by the specific embodiments described herein, but rather by the appended claims, which are intended to be construed in accordance with the following well-settled principles of claim construction: (a) Each claim should be given its broadest reasonable interpretation consistent with the specification; (b) Limitations should not be read from the specification or drawings into the claims (e.g., if the claim calls for "antenna", and the specification and drawings show a coil, the claim term "antenna" should not be limited to a coil, but rather should be construed to cover any type of antenna); (c) The words "comprising", "including", and "having" are always open-ended, irrespective of whether they appear as the primary transitional phrase of a claim or as a transitional phrase within an element or sub-element of the claim; (d) The indefinite articles "a" or "an" mean one or more; where, instead, a purely singular meaning is intended, a phrase such as "one", "only one", or "a single", will appear; (e) Words in a claim should be given their plain, ordinary, and generic meaning, unless it is readily apparent from the specification that an unusual meaning was intended; (f) an absence of the specific words "means for" connotes applicants' intent not to

16

invoke 35 U.S.C. §112 (6) in construing the limitation; (g) Where the phrase "means for" precedes a data processing or manipulation "function," it is intended that the resulting means-plus-function element be construed to cover any, and all, computer implementation(s) of the recited "function"; (h) a claim that contains more than one computer-implemented means-plus-function element should not be construed to require that each means-plus-function element must be a structurally distinct entity (such as a particular piece of hardware or block of code); rather, such claim should be construed merely to require that the overall combination of hardware/firmware/software which implements the invention must, as a whole, implement at least the function(s) called for by the claim's means-plus-function element(s); (i) a means-plus-function element should be construed to require only the "function" specifically articulated in the claim, and not in a way that requires additional "functions" which may be described in the specification or performed in the preferred embodiment(s); (j) The existence of method claims that parallel a set of means-plus-function apparatus claims does not mean, or suggest, that the method claims should be construed under 35 U.S.C. §112 (6).

We claim:

1. A method for accessing information in at least one digital logger storing data associated with input from a plurality of input channels, comprising:

at a Web server having access to said at least one digital logger, receiving a request for retrieval of stored data from a client;

retrieving stored data in accordance with the received request; and

transferring the retrieved data to the client.

2. The method of claim 1 wherein the step of retrieving stored data comprises accessing a record of an input channel made by said at least one digital logger.

3. The method of claim 2 wherein the step of retrieving stored data comprises accessing call information for a record of an input channel made by said at least one digital logger.

4. The method of claim 2 wherein the step of retrieving stored data comprises providing direct access to a record of an input channel stored in said at least one digital logger.

5. The method of claim 1 wherein the step of retrieving stored data comprises causing a process to access data stored in said at least one digital logger through a common gate interface.

6. The method of claim 1 wherein the step of retrieving stored data comprises accessing archived data at the Web server corresponding to a record of an input channel made by said at least one digital logger.

7. A method for accessing information in a digital logging system storing data associated with input from a plurality of communication channels, comprising:

receiving an information request from a user at a network server having access to a plurality of data records created by the logging system, the records corresponding to data transmitted over a communication channel; providing to the user a description of services offered by the network server;

receiving at the network server an indication of a service selected by the user;

running at the network server of a software routine causing the server to elicit from the user one or more of the following: user information, user ID, authorization level, password and payment information;

in response to the elicited user data providing access to one or more stored records created by the logging system.

US 6,870,920 B2

17

8. The method of claim 7, wherein the stored data records have unique record addresses.

9. The method of claim 7, wherein the network server is a Web server.

10. The method of claim 9, wherein the user and the Web server communicate through a web browser.

11. The method of claim 9 wherein the step of providing a description of services is performed by causing a home page associated with the logging system to be displayed.

12. The method of claim 7, wherein the network server is a file server.

13. The method of claim 7, wherein the step of providing access to one or more records comprises providing record information including one or more of: file name, date created, size of the record, audio type files in use, total associated files, and a command set code.

14. The method of claim 13 further comprising the step of providing the user with instructions for decoding playback software.

15. The method of claim 7, wherein at least some data records created by the logging system are archived at the network server.

16. A method for accessing information stored by at least one digital logger storing data associated with input from a plurality of communication channels, comprising:

at a Web server having access to said information stored by at least one digital logger over a communications network, receiving a request for retrieval of stored data from a user;

retrieving said stored data from said information in accordance with the received request; and

transferring the retrieved data to the client.

17. The method of claim 16 wherein the step of retrieving stored data comprises accessing a record of a communication channel made by said at least one digital logger.

18. The method of claim 17 wherein the step of retrieving stored data comprises accessing call information for a record of a communication channel made by said at least one digital logger.

19. The method of claim 17 wherein the step of retrieving stored data comprises providing direct access to a record of

18

a communication channel in said information stored by said at least one digital logger.

20. The method of claim 16 wherein the step of retrieving stored data comprises causing a process to access data stored in said at least one digital logger through a common gate interface.

21. The method of claim 16 wherein the step of retrieving stored data comprises accessing archived data at the Web server corresponding to a record of a communication channel made by said at least one digital logger.

22. A method for accessing information in at least one digital logger storing data from a plurality of input channels, comprising:

at a Web server having access to said at least one digital logger, receiving a request from a client for retrieval of stored data from one or more of a plurality of input channels;

retrieving data comprising said stored data in accordance with said received request; and

transferring said retrieved data to the client.

23. The method of claim 22 wherein the step of retrieving stored data comprises accessing a record of an input channel made by said at least one digital logger.

24. The method of claim 23 wherein the step of retrieving stored data comprises accessing call information for a record of an input channel made by said at least one digital logger.

25. The method of claim 24 wherein said call information comprises voice data.

26. The method of claim 23 wherein the step of retrieving stored data comprises providing direct access to a record of an input channel stored in said at least one digital logger.

27. The method of claim 22 wherein the step of retrieving stored data comprises causing a process to access data stored in said at least one digital logger through a common gate interface.

28. The method of claim 22 wherein the step of retrieving stored data comprises accessing archived data at the Web server corresponding to a record of an input channel made by said at least one digital logger.

* * * * *

EXHIBIT I



US006959079B2

(12) **United States Patent**
Elazar

(10) **Patent No.:** **US 6,959,079 B2**
(45) **Date of Patent:** **Oct. 25, 2005**

(54) **TELEPHONE CALL MONITORING SYSTEM**

(56) **References Cited**

(75) **Inventor:** **Avishai Elazar, Sunnyvale, CA (US)**

U.S. PATENT DOCUMENTS

(73) **Assignee:** **Nice Systems Ltd., Ra'anana (IL)**

5,946,375 A 8/1999 Pattison et al.
6,230,197 B1 * 5/2001 Beck et al. 709/223
6,263,049 B1 * 7/2001 Kuhn 379/32.01

(*) **Notice:** Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 14 days.

* cited by examiner

(21) **Appl. No.:** **10/367,749**

Primary Examiner—Benny Tieu

(22) **Filed:** **Feb. 19, 2003**

(74) **Attorney, Agent, or Firm**—Pearl Cohen Zedek Latzer, LLP

(65) **Prior Publication Data**

(57) **ABSTRACT**

US 2003/0147522 A1 Aug. 7, 2003

Related U.S. Application Data

(63) Continuation of application No. 09/503,479, filed on Feb. 14, 2000, now Pat. No. 6,542,602.

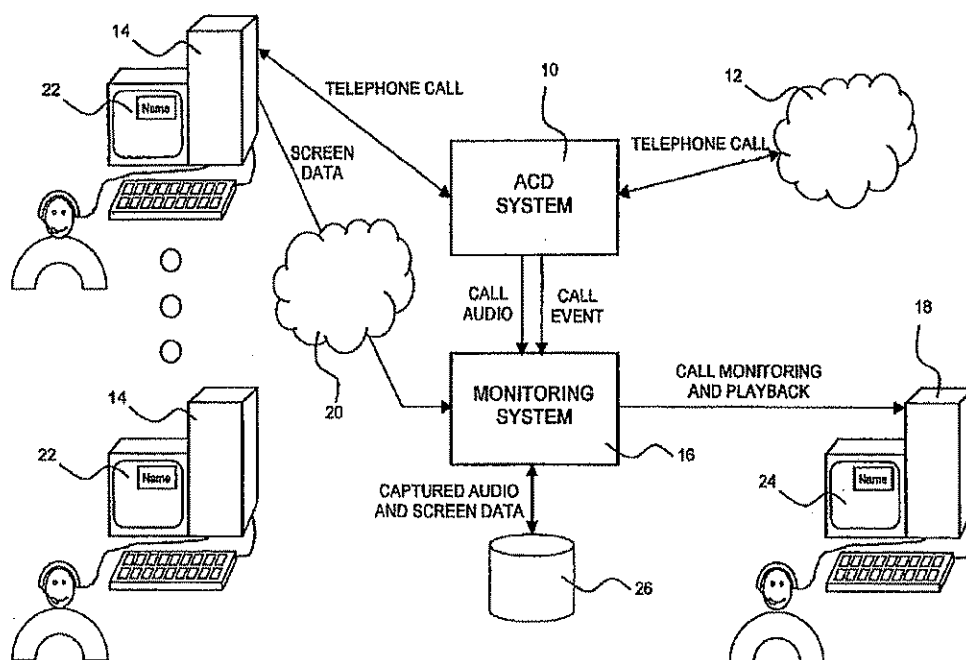
(51) **Int. Cl.**⁷ **H04M 3/00; H04M 5/00**

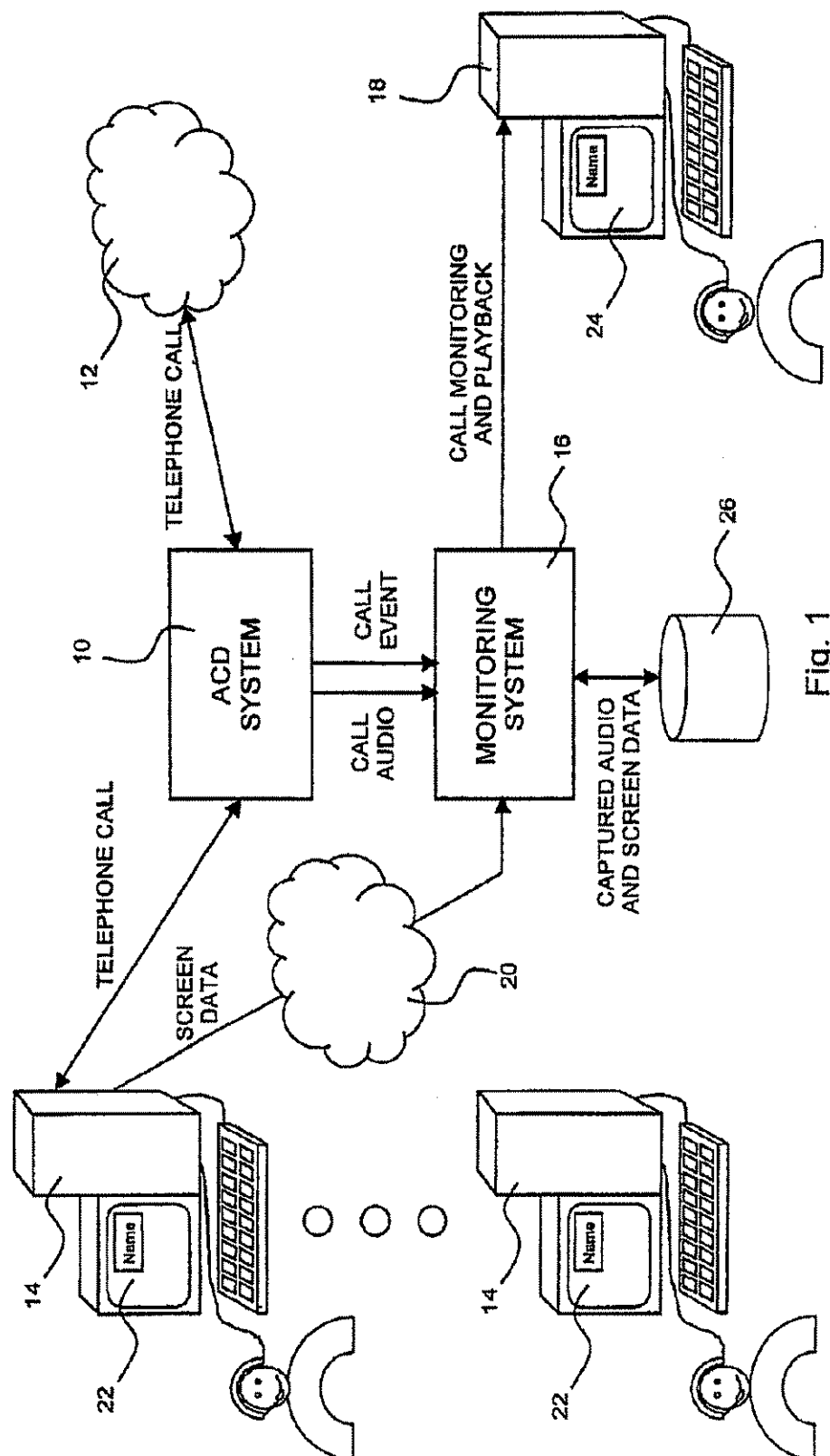
(52) **U.S. Cl.** **379/265.06**

(58) **Field of Search** 379/265.06, 265.07,
379/265.02, 266.01–266.03, 88.01–88.04,
142.01, 112.01

A monitoring system for monitoring agent telephonic interactions with customers and a method thereof is provided. The system may comprise a voice logger to receive and record audio of a telephone call received by the agent, a screen logger to receive and record video screen data associated with interactions of the agent with a computer during the telephone call and an event manager to determine whether the interactions meet at least one predefined monitoring condition.

6 Claims, 13 Drawing Sheets





U.S. Patent

Oct. 25, 2005

Sheet 2 of 13

US 6,959,079 B2

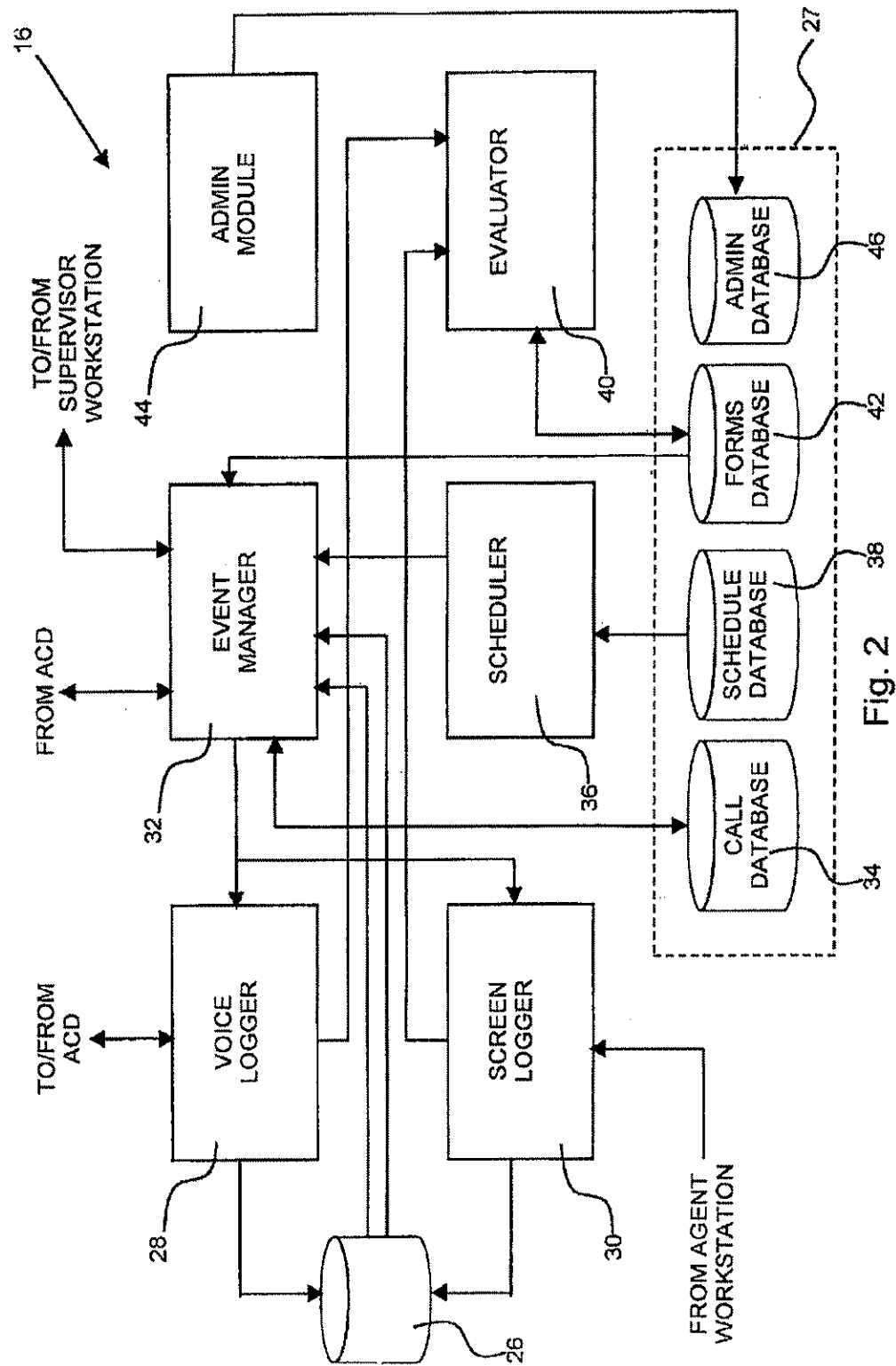


Fig. 2

U.S. Patent

Oct. 25, 2005

Sheet 3 of 13

US 6,959,079 B2

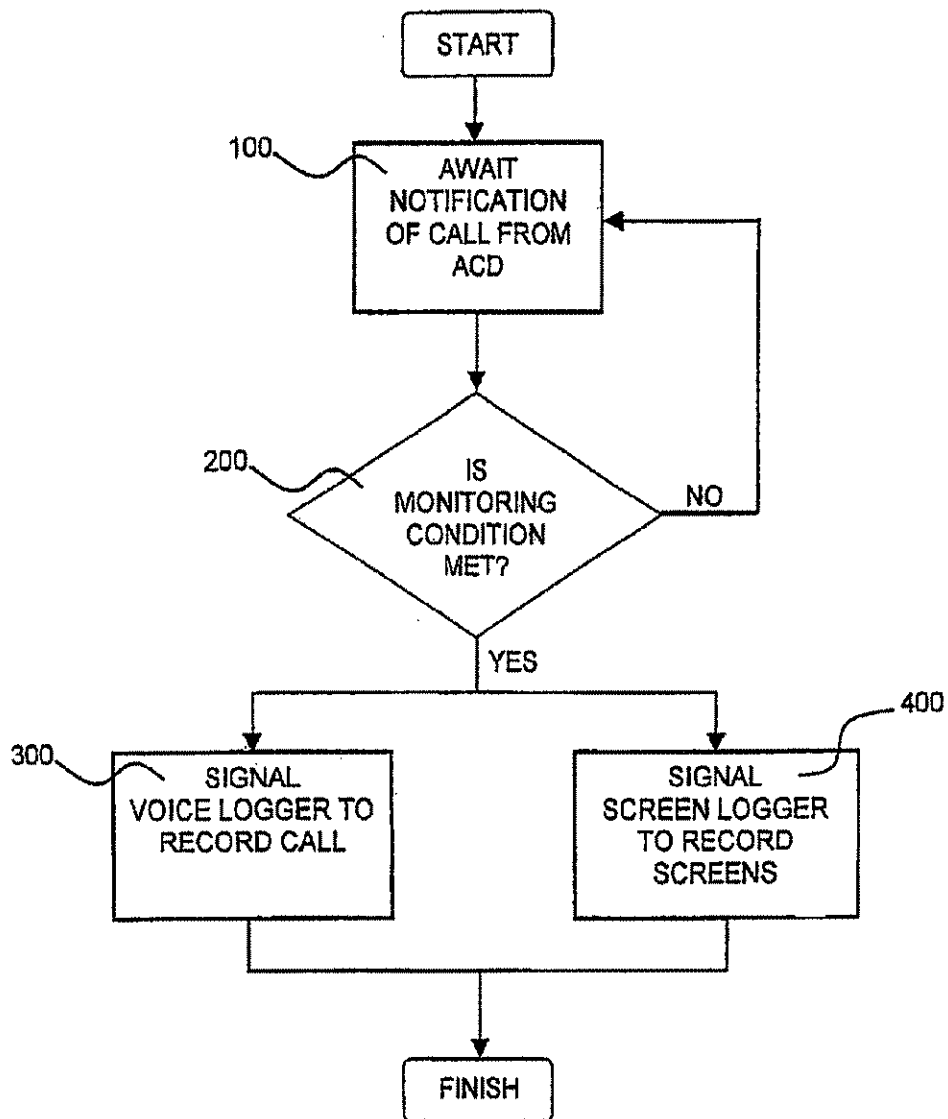


Fig. 3

U.S. Patent

Oct. 25, 2005

Sheet 4 of 13

US 6,959,079 B2

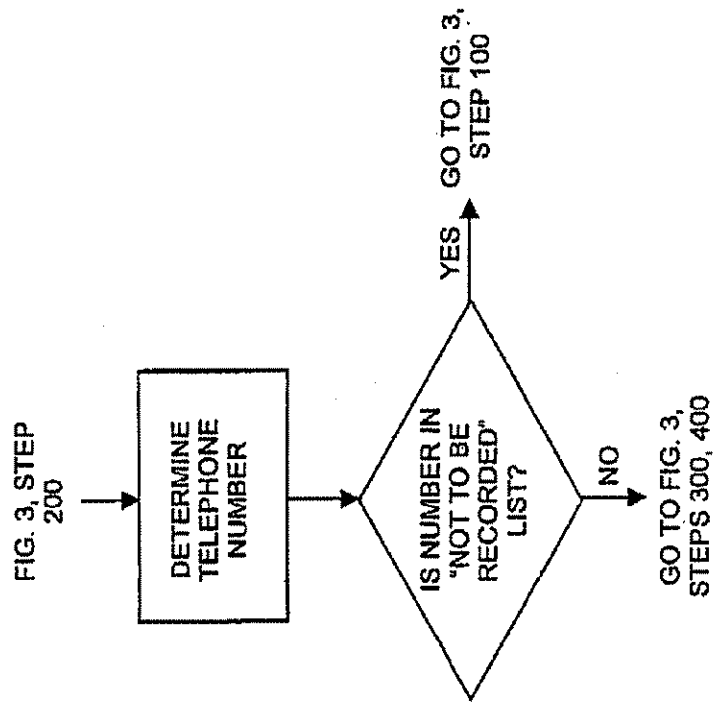


Fig. 5

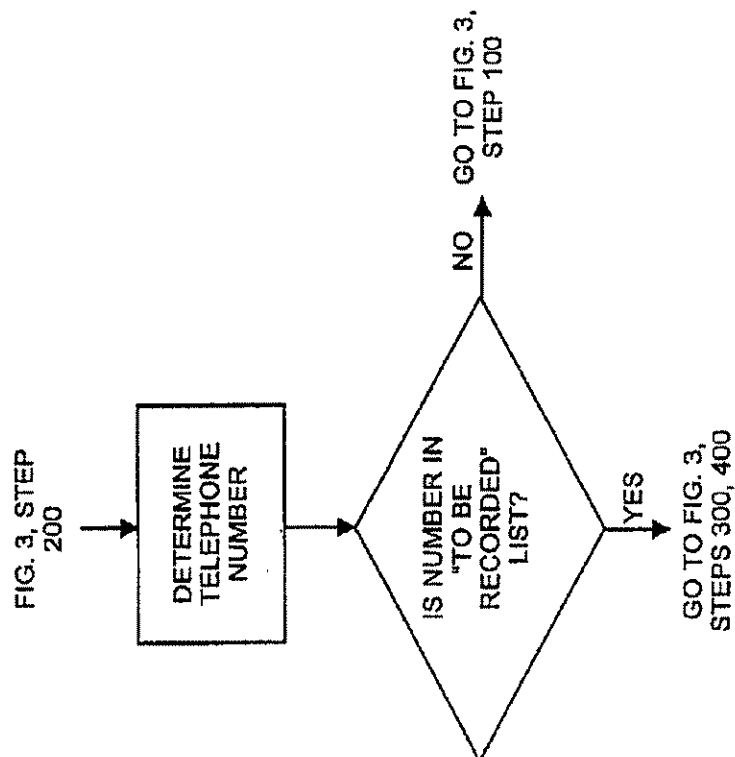


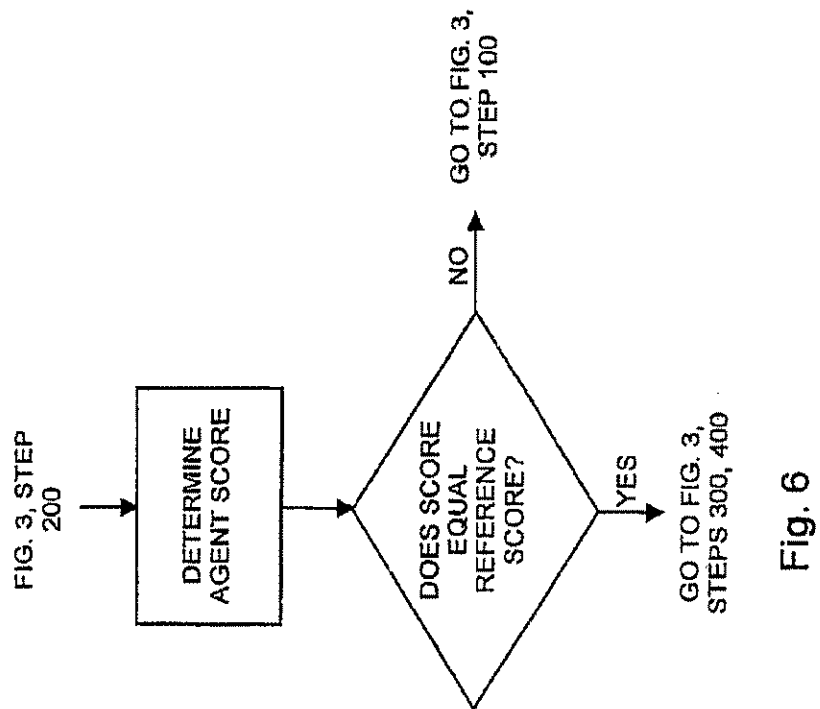
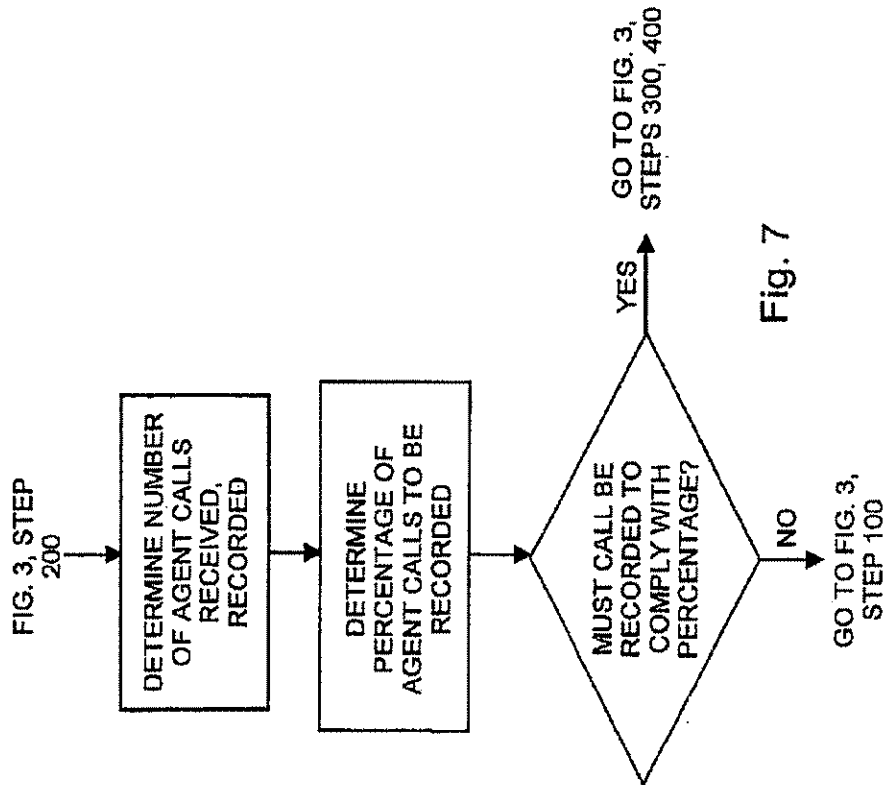
Fig. 4

U.S. Patent

Oct. 25, 2005

Sheet 5 of 13

US 6,959,079 B2

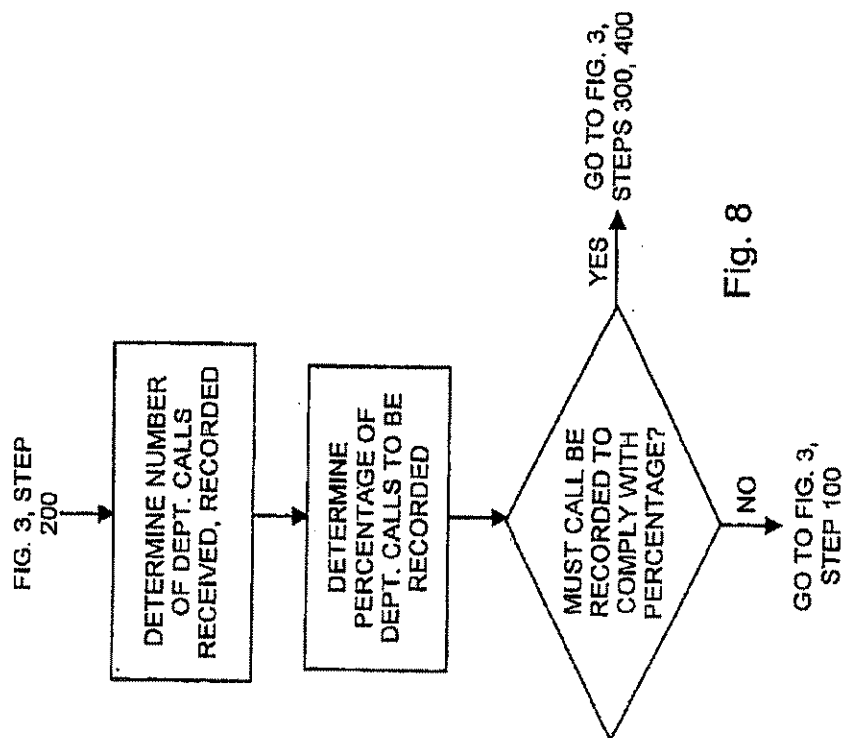
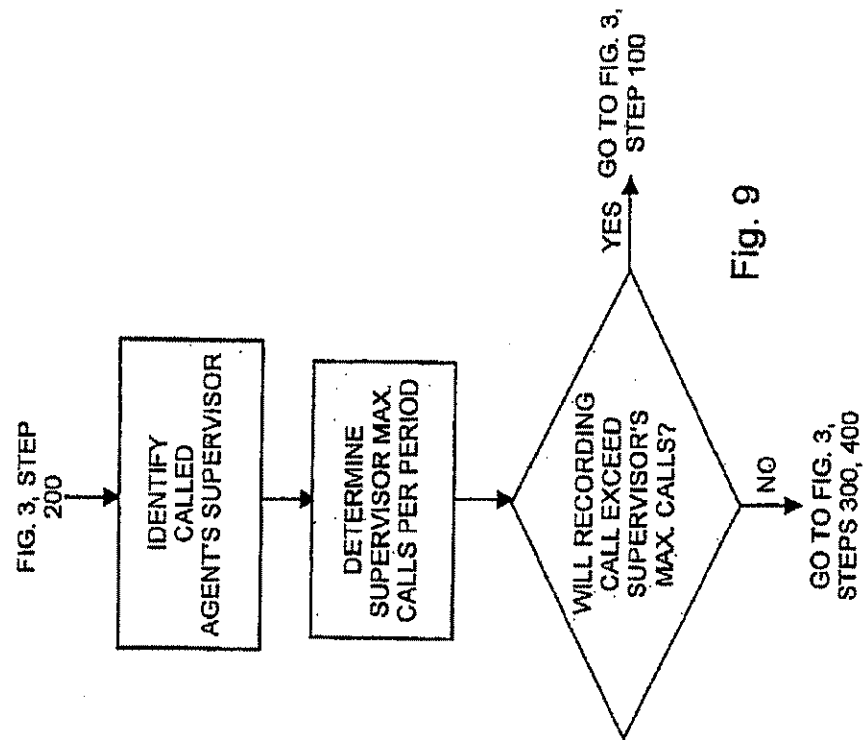


U.S. Patent

Oct. 25, 2005

Sheet 6 of 13

US 6,959,079 B2



U.S. Patent

Oct. 25, 2005

Sheet 7 of 13

US 6,959,079 B2

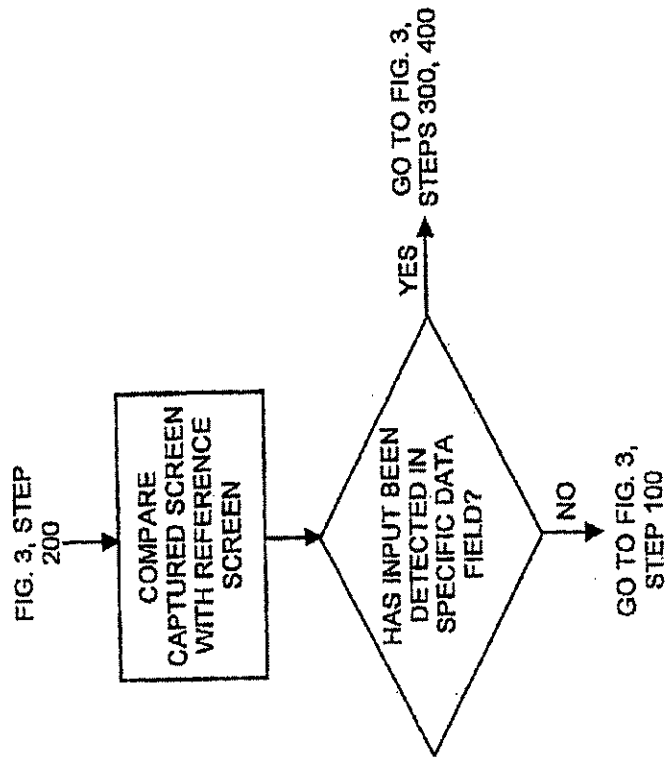


Fig. 11

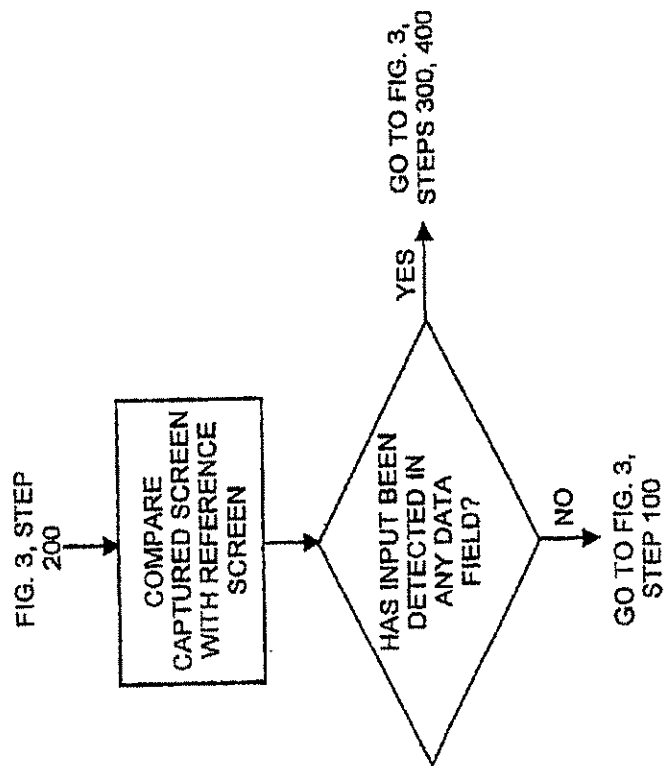


Fig. 10

U.S. Patent

Oct. 25, 2005

Sheet 8 of 13

US 6,959,079 B2

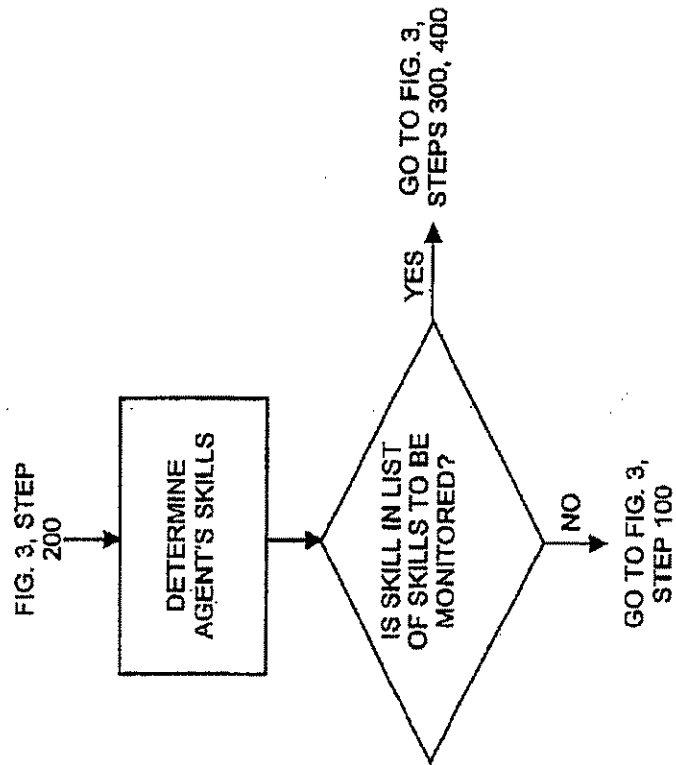


Fig. 13

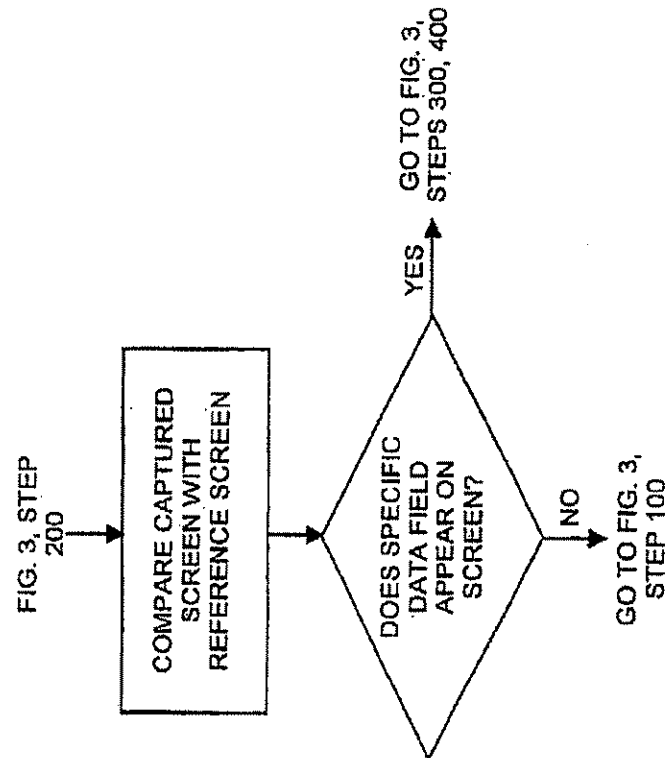


Fig. 12

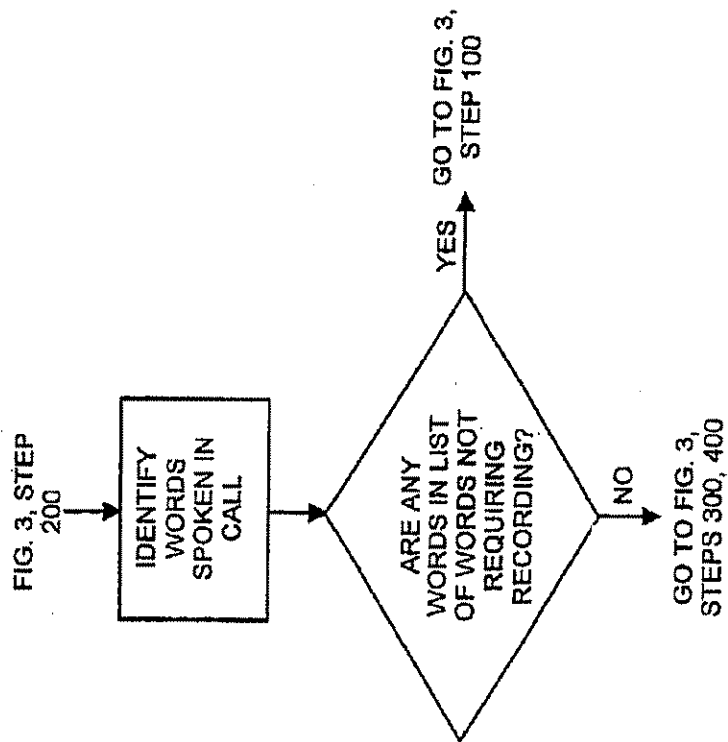


Fig. 15

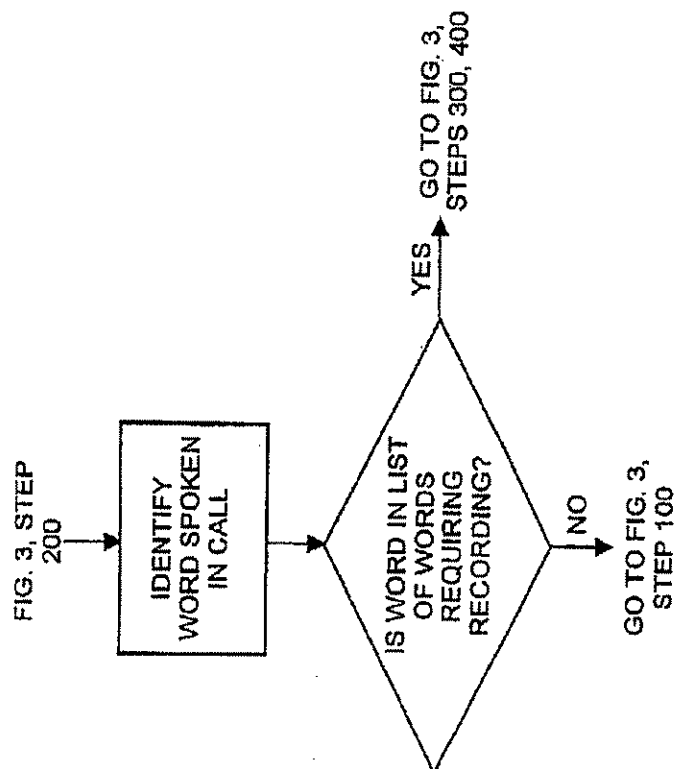


Fig. 14

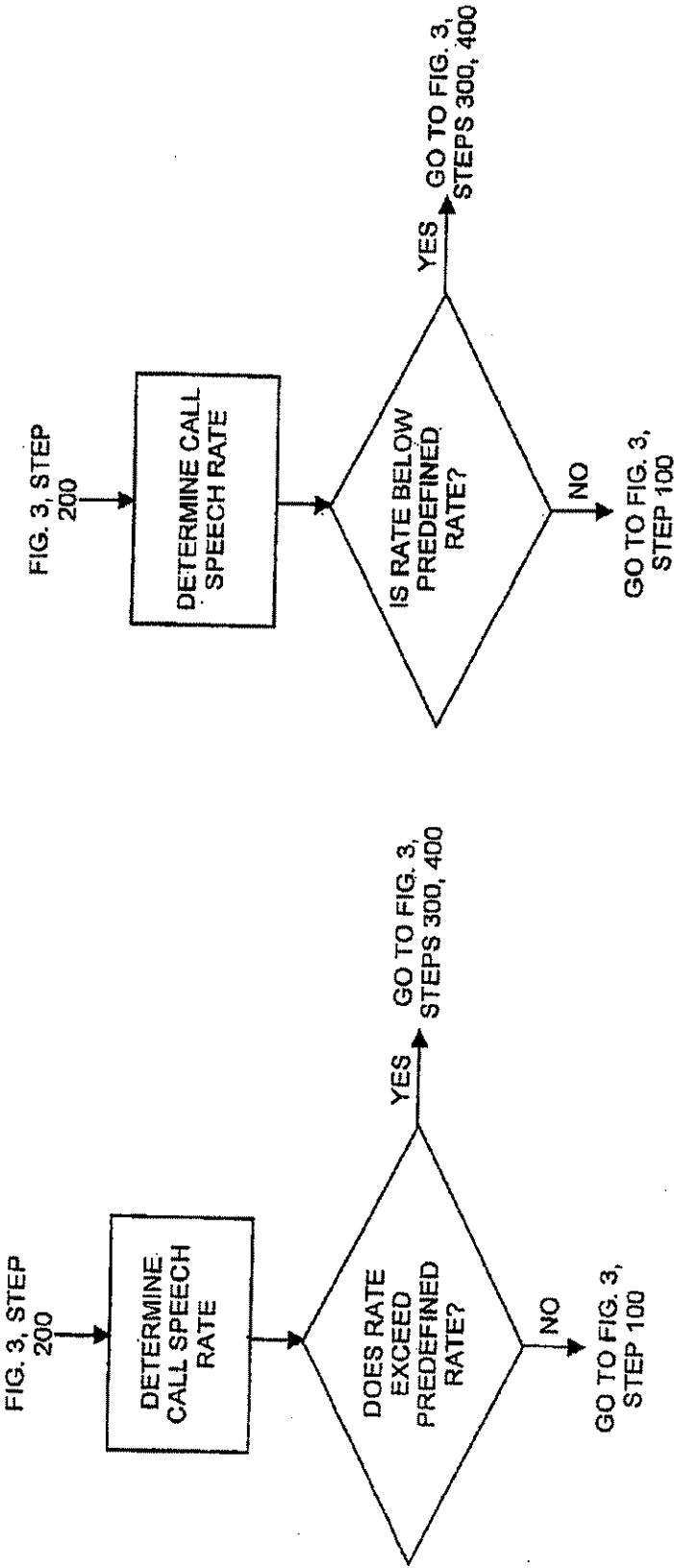


Fig. 17

Fig. 16

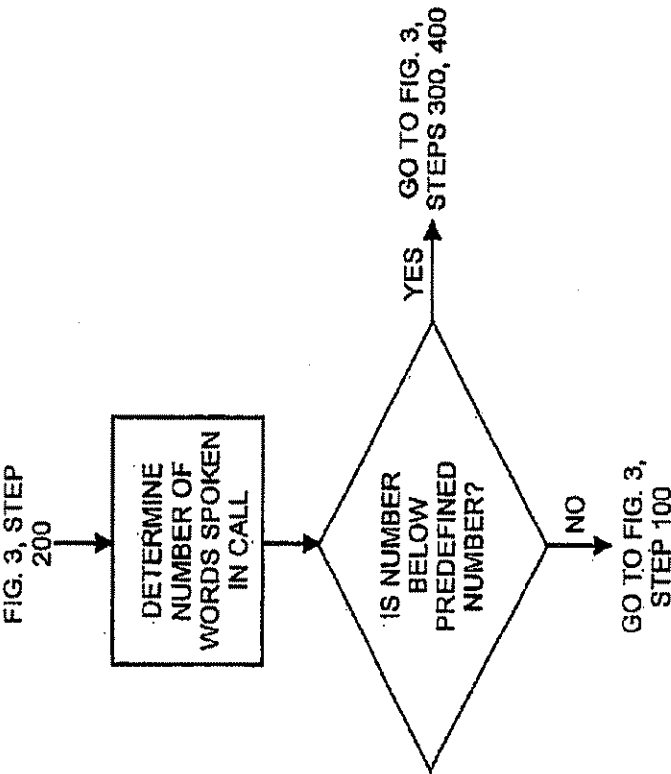


Fig. 19

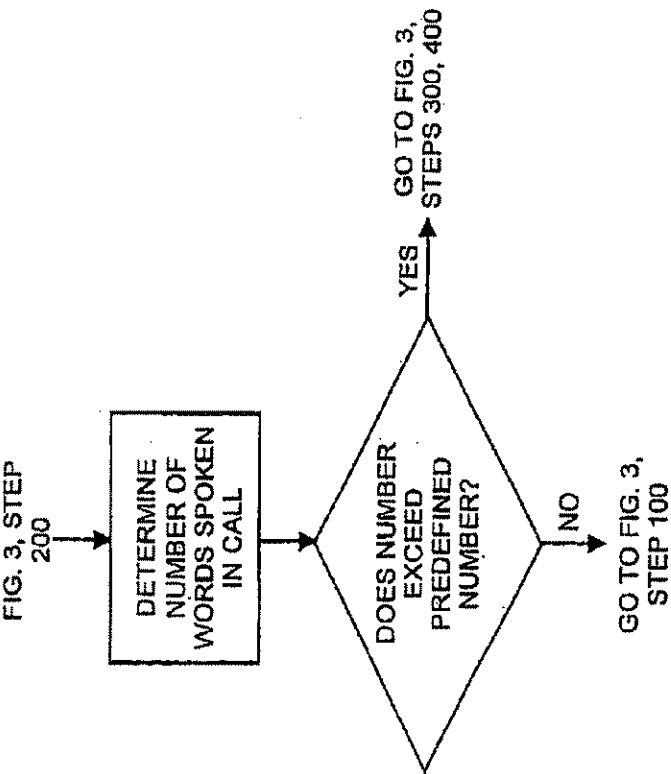
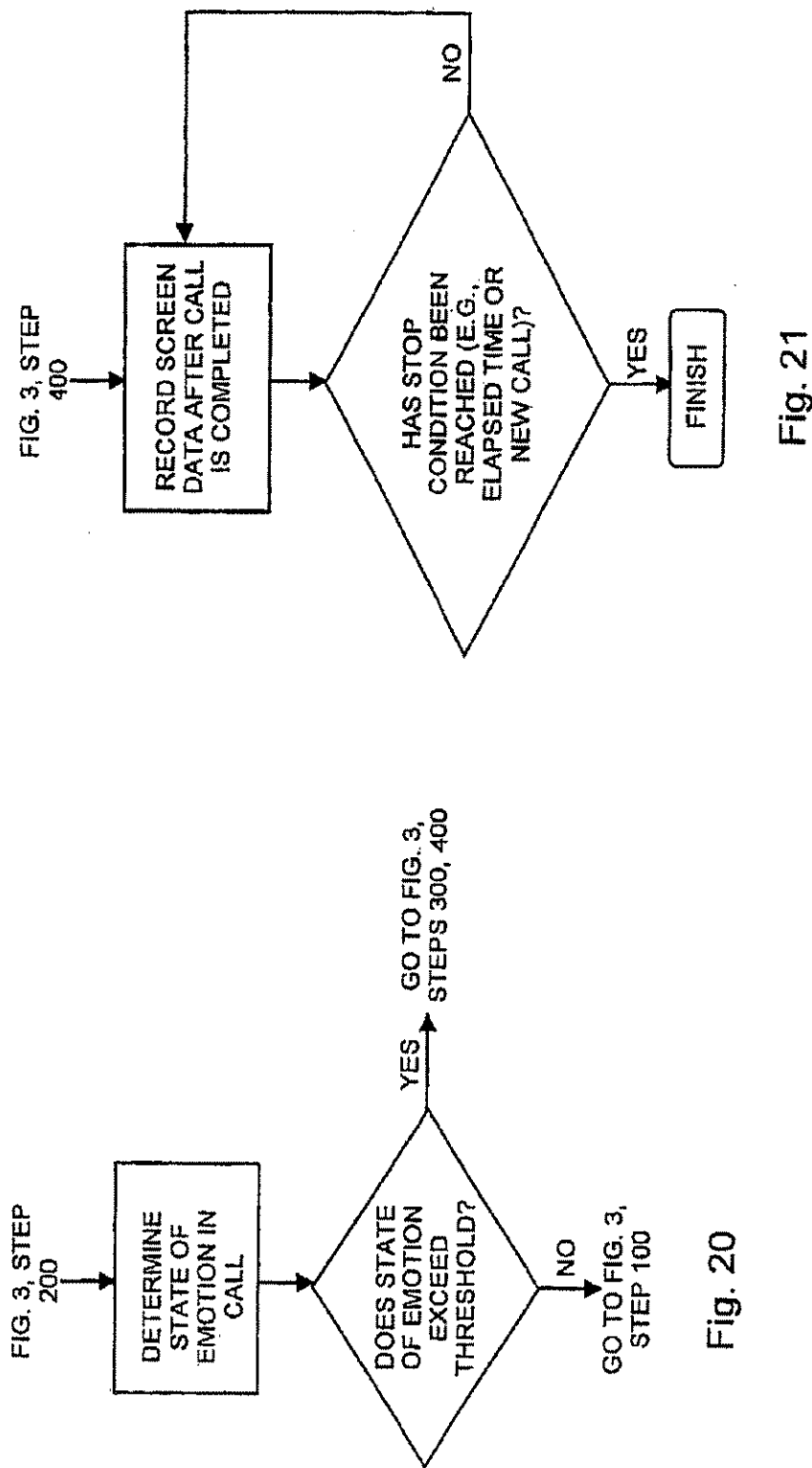


Fig. 18



U.S. Patent

Oct. 25, 2005

Sheet 13 of 13

US 6,959,079 B2

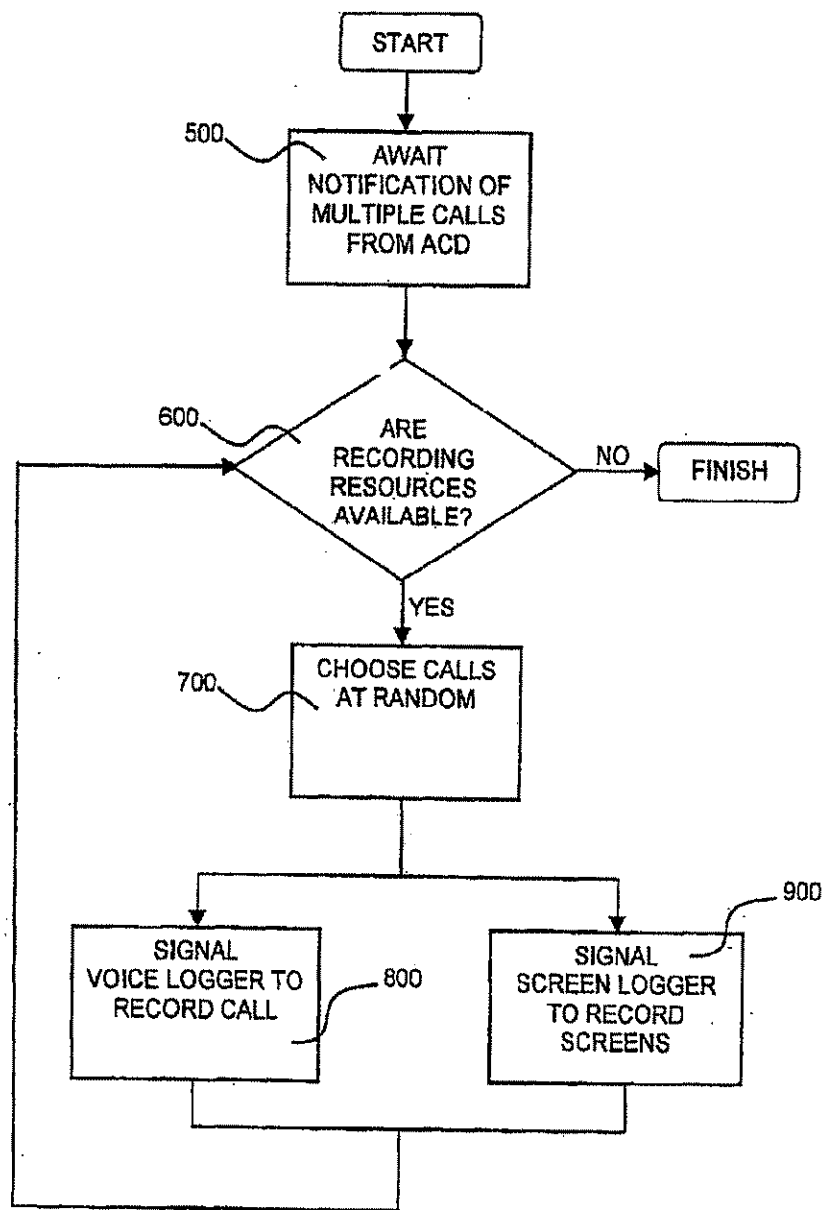


Fig. 22

US 6,959,079 B2

1

TELEPHONE CALL MONITORING SYSTEM**CROSS REFERENCE TO RELATED APPLICATIONS**

This application is a continuation application of patent application Ser. No. 09/503,479 filed on Feb. 14, 2000 now U.S. Pat. No. 6,542,602.

FIELD OF THE INVENTION

The present invention is related to telephone call monitoring systems in general, and in particular to apparatus and methods for logging telephone calls according to non-time-based scheduling criteria.

BACKGROUND OF THE INVENTION

Telephone call monitoring systems are increasingly being used by businesses to monitor the effectiveness of agents who receive telephone calls. In prior art telephone call monitoring systems data are typically collected on each incoming call to the system. This data consists of a log of events occurring in the system over time for an incoming call. Typical logged data elements are receipt of call, call offered to an application, call presented to an agent or an agent group, call handled or abandoned and length of call. The data representing these data elements are then processed to generate reports for use by management or supervisory personnel. The data may be organized in any number of ways, such as by agent, telephone trunk, agent groups and the like. Based on this information, management and supervisory personnel are able to evaluate an agent's telephone call activity and take corrective action where an agent's performance falls below acceptable norms.

Unfortunately, prior art telephone call monitoring systems are limited in their ability to discriminate between different telephone calls and agents based on criteria other than simple scheduling imperatives and rely almost entirely on human intervention to evaluate the quality of service provided by an agent. Prior art telephone call monitoring systems that provide for telephone call recording typically record either all telephone calls received by an agent or record telephone calls according to a schedule.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be understood and appreciated more fully from the following detailed description taken in conjunction with the appended drawings in which:

FIG. 1 is a simplified block diagram of an exemplary Automatic Call Distribution (ACD) system connected to an external telephonic network and agent telephonic workstations in which a telephone call monitoring system, constructed and operative in accordance with a preferred embodiment of the present invention, may be advantageously implemented;

FIG. 2 is a simplified block diagram of a telephone call monitoring system, constructed and operative in accordance with a preferred embodiment of the present invention;

FIG. 3 is a simplified flowchart illustration of a method of operation of the telephone call monitoring system of FIG. 2, the method being operative in accordance with a preferred embodiment of the present invention;

FIGS. 4-20 are simplified flowchart illustrations of implementations of the monitoring condition step 200 of FIG. 3, operative in accordance with multiple preferred embodiments of the present invention;

2

FIG. 21 is a simplified flowchart illustration of an implementation of recording step 400 of FIG. 3, operative in accordance with a preferred embodiment of the present invention; and

FIG. 22 is a simplified flowchart illustration of a method of operation of monitoring system 16 of FIG. 1, the method being operative in accordance with a preferred embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION**Notation Used Throughout**

The following notation is used throughout this document.

Term	Definition
ACD	Automatic Call Distributor. A telephony switch capable of managing the automatic distribution of incoming calls to agents or sales representatives based on predefined call allocation algorithms. ACDs are usually found in call centers.
ANI	Automatic Number Identification. The telephone number of the calling party. This data is provided to the called party by the PSTN.
CTI	Computer Telephony Integration. A set of applications that integrates the telephony resources and computerized resources of an organization in order to provide added value services.
CTI link	An external port of a telephony switch via which an external computer can get information regarding calls managed by the switch and can control the operation of the switch as well.
DNIS	Dialed Number Identification Service. A service that identifies the called telephone number.
DTMF	Dual Tone Multiple Frequency. A set of predefined combinations of tones generated by pressing the keys of a telephone set.
IVR	Interactive Voice Response. A computer-based product connected to a switch. IVR product enables a remote caller to pass or to retrieve information to or from a database or other computerized platform connected to the switch.
PSTN	Public Switched Telephone Network.
QA	Quality Assurance.

Reference is now made to FIG. 1 which is a simplified block diagram of an exemplary prior art Automatic Call Distribution system, herein referred to as ACD 10, connected to an external telephonic network 12 and a plurality of agent telephonic workstations 14 in which a monitoring system 16, constructed and operative in accordance with a preferred embodiment of the present invention, may be advantageously implemented. ACD 10 routes incoming telephone calls received via network 12 to a plurality of agent telephonic workstations 14. As will be readily apparent to those skilled in the art, ACD 10 and telephonic workstations 14 may be any of a number of different, commercially-available systems or devices used in ACD systems.

Monitoring system 16 is preferably in communication with ACD 10 for receiving notifications of telephone calls that have been routed by ACD 10 to agent telephonic workstations 14 and for receiving therefrom any audio portion of the telephone calls. Monitoring system 16 is also preferably in communication with agent telephonic workstations 14, either directly or via a computer network 20, such as a local area network (LAN), for receiving screen data captured from a display 22 of workstation 14 before, during, or after a telephone call is received at workstation 14, typically using any conventional screen data capture means assembled with workstation 14.

Monitoring system 16 preferably stores captured audio and screen data to one or more storage media 26 and

US 6,959,079 B2

3

provides captured audio and screen data to one or more supervisor workstations 18 either in real-time or later in a playback mode where audio, screen data, and other data may be monitored separately or simultaneously. Supervisor workstation 18 preferably includes a display 24 for displaying information relating to the operation of ACD 10 and to the performance of an agent as is described in greater detail hereinbelow.

Additional reference is now made to FIG. 2, which is a simplified block diagram of monitoring system 16 of FIG. 1, constructed and operative in accordance with a preferred embodiment of the present invention. Monitoring system 16 preferably comprises a voice logger 28 which receives telephone call audio from ACD 10 and records the audio. Voice logger 28 is typically connected to ACD 10 via an ACD input or output channel or via a LAN audio channel. Voice logger 28 may also provide audio playback to ACD 10. A screen logger 30 is also provided which receives video screen data from workstations 14 and records the captured screen data. Monitoring system storage 27 is also provided, comprising a call database 34, a schedule database 38, a forms database 42, and an admin database 46.

An event manager 32 receives notifications from ACD 10 of telephone calls that have been routed to workstations 14 and preferably logs information regarding the calls received from ACD 10 in call database 34. Such information typically includes the outside party's telephone number, the locations of the telephone call recordings in storage 26, the time the call was made or received, and the duration of the call. Event manager 32 may then control voice logger 28 and screen logger 30 to begin recording audio and screen data of a telephone call for which an event notification has been received. Event manager 32 is typically connected to ACD 10 via a Computer Telephony Integration (CTI) port or via an ACD management port. Event manager 32 may provide audio and screen data associated with a particular telephone call as soon as it is switched to supervisor workstation 18 for real-time monitoring, or may receive playback requests from supervisor workstation 18, in response to which event manager 32 retrieves the requested audio and/or screen data from storage media 26 and provides the audio and/or screen data to workstation 18 for output.

A scheduler 36 provides event manager 32 with scheduling information stored in schedule database 38. The scheduling information may include:

1. how often to monitor a particular agent or all agents;
2. how often to monitor a particular agent group (where an agent in the group might then be selected at random for monitoring);
3. a percentage of calls to be monitored for a particular agent, all agents, an agent group, or department;
4. how many calls a particular supervisor or all supervisors are to monitor in a given period.

The scheduling information may also include non-temporal "scheduling" criteria such as

1. telephone numbers of customers who should or should not be monitored;
2. specific data fields on display 22 (FIG. 1) that, when data are entered into the fields, signal that the call is to be monitored;
3. a table of agent scores and the monitoring frequency for each score level;
4. a table of agent skills and the monitoring frequency for each skill;
5. words that, when spoken during a telephone call, signal that the call is to be monitored (e.g., obscenities);

4

6. words that, when not spoken during a telephone call, signal that the call is to be monitored (e.g., "please" and "thank you");

7. a table of speech rates and the monitoring frequency for each speech rate;

8. a table of word counts and the monitoring frequency for different word counts; and

9. CTI data, including business information gathered regarding the calling or called party.

Event manager 32 may use the information in database 38 to make scheduling determinations such as whether a telephone call received from or made to a particular telephone number, such as may be determined through Automatic Number Identification (ANI) or Dialed Number Identification Service (DNIS), is due to be recorded, whether a particular agent is due for monitoring, whether the supervisor is due to monitor any or a particular call or agent, or whether other calling functions, such as "call transfer" or "hold" as may be determined through DTMF activation signals, require monitoring.

An evaluator 40 preferably provides a means for designing evaluation forms and for storing forms and evaluation data in forms database 42. Evaluation forms may be completed on-screen by a supervisor via workstation 18 or by an agent via workstation 14 for such purposes as evaluating agent performance or providing a customer interaction debriefing. Evaluator 40 may also receive audio and/or screen data from voice logger 28 and screen logger 30 and perform automated evaluations in accordance with preprogrammed algorithms. Such evaluations may include determining what words are spoken during a telephone call and speech rates using Digital Signal Processing (DSP) and speech recognition technologies well known in the art. Event manager 32 preferably accesses evaluation data stored in database 42 to support monitoring decisions as described hereinabove.

An administration module 44 provides for the definition of agents, supervisors, and other users and groups who may access various aspects of monitoring system 16 as well as security rules for such access. These definitions are stored in administration database 46.

Although not particularly shown in FIG. 2, any of voice logger 28, screen logger 30, event manager 32, scheduler 36, evaluator 40, and admin module 42 may access any of the information in databases 34, 38, 42, and 46. For example, scheduler 36 may access admin database 46 to detect changes in agent detail and modify scheduling information accordingly.

Reference is now made to FIG. 3, which is a simplified flowchart illustration of a method of operation of monitoring system 16 of FIG. 1, the method being operative in accordance with a preferred embodiment of the present invention. In the method of FIG. 3 a notification of a customer telephone call is received from ACD 10 indicating that a call is currently in progress with a particular agent (step 100). Event manager 32 (FIG. 2) then determines whether the telephone call is to be monitored by determining whether at least one predefined monitoring condition is true (step 200). If the condition is true, event manager 32 signals voice logger 28 to record some or all of the audio of the telephone call (step 300) and screen logger 30 to record in synchronicity with the audio recording of voice logger 28 some or all of the agent's interactions with the agent's workstation during the telephone call, preferably in the form of screen data captures (step 400).

A portion of the audio and/or screen data may be pre-recorded prior to and in support of performing step 200, in

US 6,959,079 B2

5

which the monitoring condition is tested. Under such circumstances, the signals to record referred to in steps 300 and 400 may be understood as directives to continue to record once the satisfaction of the monitoring condition has been established, or, if step 200 is carried out after the entire call has been recorded, as directives to retain the recording already made. Where a first portion of the telephone call was pre-recorded prior to performing step 200, it may be combined with any portions of the telephone call recorded after satisfaction of the monitoring condition has been established.

Reference is now made to FIG. 4 which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment, of the present invention. In the method shown the determining step comprises determining the telephone number from which the call has been made. The monitoring condition is satisfied if the telephone number or a portion thereof is in a database of telephone numbers whose calls are to be recorded, the information typically maintained in schedule database 38 (FIG. 2). An example of a telephone number portion may be 212-xxx-xxxx, indicating that all calls from the 212 area code are to be recorded, or 212-605-xxxx, indicating that only area code 212 calls from exchange 605 are to be recorded.

Reference is now made to FIG. 5, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining the telephone number from which the call has been made. The monitoring condition is satisfied if the telephone number is not in a database of telephone numbers whose calls are not to be recorded, the information typically maintained in schedule database 38 (FIG. 2). An example of a telephone number portion may be 212-xxx-xxxx, indicating that all calls outside the 212 area code are to be recorded, or 212-605-xxxx, indicating that calls outside area code 212 and exchange 605 are to be recorded.

Reference is now made to FIG. 6 which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises retrieving a score for the agent, the information typically maintained in forms database 42. The monitoring condition is satisfied if the score is equal to a predetermined reference score that warrants monitoring of the call.

Reference is now made to FIG. 7, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises retrieving the number of calls which the agent has received and the number of the calls which have been recorded, both of which are typically maintained in calls database 34, and a percentage of the agent's calls which are to be recorded, the information typically maintained in schedule database 38. The monitoring condition is satisfied if the call currently in progress must be recorded in order to comply with the percentage of the agent's calls which are to be recorded.

Reference is now made to FIG. 8 which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining the department to which the agent belongs, the information typically main-

6

tained in admin database 46, retrieving the number of calls which the department has received and the number of the calls which have been recorded, both of which are typically maintained in calls database 34 or may be calculated therefrom, and determining the percentage of the department's calls which are to be recorded, the information typically maintained in schedule database 38. The monitoring condition is satisfied if the call currently in progress must be recorded in order to comply with the percentage of the department's calls, which are to be recorded.

Reference is now made to FIG. 9, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises associating the agent with a supervisor, with the agent's supervisor typically maintained in admin database 46. The number of calls which the supervisor is to monitor in a given time period is then determined, the information typically maintained in schedule database 38. The monitoring condition is satisfied if recording the call currently in progress and within the time period will not cause the number of calls to be exceeded.

Reference is now made to FIG. 10, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining whether input has been entered into any data field appearing on a display of the computer. Evaluator 40 may provide this information by comparing a captured screen with a reference screen into which no data has been entered and storing an input indicator in forms database 42.

Reference is now made to FIG. 11, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining whether input has been entered into at least one specific data field appearing on a display of the computer. Evaluator 40 may provide this information by comparing a captured screen with a reference screen into which no data has been entered into the specific field and storing an input indicator in forms database 42.

Reference is now made to FIG. 12 which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining whether at least one specific data field appears on a display of the computer. Evaluator 40 may provide this information by comparing a captured screen with a reference screen containing the specific field and storing a field presence indicator in forms database 42.

Reference is now made to FIG. 13, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining the agent's skills, the information typically maintained in admin database 46. The monitoring condition is satisfied if a possessed skill is among a predetermined list of skills that warrant monitoring of the call.

Reference is now made to FIG. 14, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises identifying at least one word

US 6,959,079 B2

7

spoken during the telephone call, the information typically maintained in forms database 42. The monitoring condition is satisfied if the word is in a database of words, which require the recording of telephone calls. Such a word may be "supervisor," possibly indicating a request to speak with a supervisor, or an obscenity.

Reference is now made to FIG. 15, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises identifying a plurality of words spoken during the telephone call, the information typically maintained in forms database 42. The monitoring condition is satisfied if one or more "good" words as found in a database of words are not present the plurality of words. Such a word may be "please" or "thank you," the absence of which might indicate a service deficiency, and, therefore, a need to record.

Reference is now made to FIG. 16, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining a speech rate of words spoken during the telephone call, the information typically maintained in forms database 42. The monitoring condition is satisfied if the speech rate exceeds a predefined speech rate, possibly indicating that the agent or the customer is talking too fast.

Reference is now made to FIG. 17, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining a speech rate of words spoken during the telephone call, the information typically maintained in forms database 42. The monitoring condition is satisfied if the speech rate is slower than a predefined speech rate, possibly indicating that the agent or the customer is talking too slowly.

Reference is now made to FIG. 18, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining the number of words spoken during the telephone call, the information typically maintained in forms database 42. The monitoring condition is satisfied if the number of words exceeds a predefined number of words, possibly indicating that the call may involve a relatively complex issue.

Reference is now made to FIG. 19, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining the number of words spoken during the telephone call, the information typically maintained in forms database 42. The monitoring condition is satisfied if the number of words is less than a predefined number of words, possibly indicating that the call may not be progressing satisfactorily.

Reference is now made to FIG. 20, which is a simplified flowchart illustration of an implementation of determining step 200 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the determining step comprises determining the state of emotion present during the telephone call using conventional means, such as the TRUSTECH system, commercially available from Truster, Ltd. of Israel. The data relating to telephone call emotion levels may be maintained in forms

8

database 42. The monitoring condition is satisfied if the detected state of emotion exceeds a certain emotion threshold as measured by the emotion detecting means.

Reference is now made to FIG. 21, which is a simplified flowchart illustration of an implementation of recording step 400 of FIG. 3, operative in accordance with a preferred embodiment of the present invention. In the method shown the recording step comprises recording at least a portion of the agent's interactions with the computer after completion of the telephone call until a stop condition is reached. The stop condition may be an elapsed period of time and/or the receipt of a notification subsequent to the notification received in step 100 that a customer telephone call is currently in progress with the agent.

Reference is now made to FIG. 22 which is a simplified flowchart illustration of a method of operation of monitoring system 16 of FIG. 1, the method being operative in accordance with a preferred embodiment of the present invention. In the method of FIG. 22 multiple notifications are received from ACD 10 indicating that multiple calls are currently in progress with multiple agents (step 500). Event manager 32 (FIG. 2) then determines whether recording resources are currently available for recording thereto (step 600). If recording resources are currently available, event manager 32 signals voice logger 28 to select any of the telephone calls at random (step 700) and record some or all of the audio of the selected calls (step 800) and screen logger 30 to record in synchronicity with the audio recordings of voice logger 28 some or all of the selected agent's interactions with the agent's workstation during the telephone call, preferably in the form of screen data captures (step 900). Recording preferably continues until the recording resources are no longer currently available for recording thereto.

It is appreciated that any of the methods described hereinabove may be carried out after one or more of the telephone calls to be evaluated have been fully recorded. In such circumstances a telephone call is pre-recorded without regard to whether or not a particular monitoring condition was satisfied. At a later time the telephone call is evaluated to determine whether the telephone call is to be retained by determining whether it meets a monitoring condition. The monitoring condition thus serves as a retention condition which, if satisfied, results in the pre-recorded telephone call being retained. The pre-recorded telephone call is then tagged with a retention indicator to indicate that the call is to be retained. The telephone call may also be tagged with a release indicator indicating when the retention indicator is to be removed, such as by specifying a predefined period of time after which said retention indicator is to be removed, or by specifying that the retention indicator is to be removed upon conclusion of an evaluation of the telephone call or after a certain period of time thereafter. The tagging with the retention indicator and/or the release indicator may be done subsequent to performing an evaluation of said telephone call, such as by a supervisor. The release indicator may also indicate a storage threshold such that the retention indicator is removed after the storage threshold of recorded telephone calls is exceeded.

The methods and apparatus disclosed herein have been described without reference to specific hardware or software. Rather, the methods and apparatus have been described in a manner sufficient to enable persons having ordinary skill in the art to readily adapt commercially available hardware and software as may be needed to reduce any of the embodiments of the present invention to practice without undue experimentation and using conventional techniques.

US 6,959,079 B2

9

While the present invention has been described with reference to a few specific embodiments, the description is intended to be illustrative of the invention as a whole and is not to be construed as limiting the invention to the embodiments shown. It is appreciated that various modifications may occur to those skilled in the art that, while not specifically shown herein, are nevertheless within the true spirit and scope of the invention.

What is claimed is:

1. A monitoring system for monitoring interactions of an agent with customers comprising:
 - a voice logger to receive and record audio of a telephone call of said agent;
 - a screen logger to receive and record video screen data associated with interactions of said agent with a computer during the telephone call; and
 - an event manager to determine whether said interactions with the computer during the telephone call meet at least one predefined monitoring condition.
2. The monitoring system of claim 1, wherein said event manager is able to instruct said voice logger to begin

10

recording an audio portion of said telephone call when said monitoring condition is satisfied.

3. The monitoring system of claim 1, wherein said event manager is able to instruct said voice logger to begin recording of an audio portion of said telephone call and to instruct said screen logger to begin recording generally in synchronicity with said voice logger at least a portion of said video screen data when said monitoring condition is satisfied.

4. The monitoring system of claim 1 further comprising: a storage media to store at least a portion of recorded audio of said telephone call and recorded video screen data associated with said telephone call.

5. The monitoring system of claim 1, further comprising: an evaluator coupled to said voice logger and to said screen logger to enable design of evaluation forms.

6. The monitoring system of claim 5, wherein said evaluator is able to perform automated evaluations based on predefined programming.

* * * * *

EXHIBIT J



US007010109B2

(12) **United States Patent**
Gritzer et al.

(10) **Patent No.:** **US 7,010,109 B2**
(45) **Date of Patent:** **Mar. 7, 2006**

(54) **DIGITAL RECORDING OF IP BASED
DISTRIBUTED SWITCHING PLATFORM**

(75) Inventors: **Hagay Gritzer**, Maccabim (IL); **Ilan Freedman**, Petach Tikva (IL); **Ilan Yosef**, Pardesiia (IL); **Danny Shporer**, Rchovot (IL)

(73) Assignee: **Nice Systems Ltd.**, Raanana (IL)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **10/906,962**

(22) Filed: **Mar. 14, 2005**

(65) **Prior Publication Data**

US 2005/0123115 A1 Jun. 9, 2005

Related U.S. Application Data

(63) Continuation of application No. 10/111,767, filed as application No. PCT/IL01/00805 on Aug. 28, 2001.

(60) Provisional application No. 60/228,124, filed on Aug. 28, 2000.

(51) **Int. Cl.**
H04M 3/42 (2006.01)

(52) **U.S. Cl.** **379/202.01**; 379/203.01;
379/265.06; 379/265.09

(58) **Field of Classification Search** 348/14.08,
348/14.09, 14.1; 370/260, 261; 379/93.21,
379/158, 202.01, 205.01, 265.02, 265.09;
709/204

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,099,510 A 3/1992 Blinken et al.
5,101,402 A 3/1992 Chiu et al.
5,559,875 A 9/1996 Bieselin et al.

5,710,591 A 1/1998 Bruno et al.
5,717,879 A 2/1998 Moran et al.
5,719,786 A 2/1998 Nelson et al.
5,764,901 A 6/1998 Skarbo et al.
5,787,253 A 7/1998 McCreery et al.
5,841,977 A 11/1998 Ishizaki et al.
5,893,053 A 4/1999 Trueblood
5,963,913 A 10/1999 Henneuse et al.
5,978,835 A 11/1999 Ludwig et al.

(Continued)

FOREIGN PATENT DOCUMENTS

WO WO 99/46702 9/1999

(Continued)

OTHER PUBLICATIONS

Toshiaki Koyama et al, Personal Multimedia Communication Systems, 8297 Hitachi Review 44 (1995) Aug., No. 4, Tokyo, Japan, p. 207-212.

(Continued)

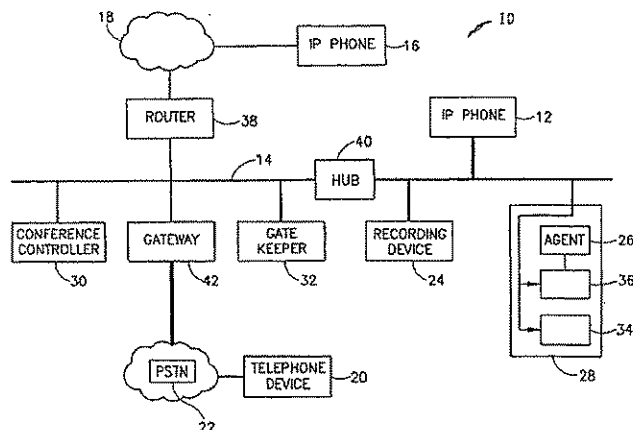
Primary Examiner—Bing Q. Bui

(74) *Attorney, Agent, or Firm*—Hershkovitz & Associates;
Abe Hershkovitz

(57) **ABSTRACT**

A system and method for recording and/or otherwise monitoring IP multimedia sessions. The present invention features a recording and/or monitoring device, referred to hereinafter as "a recording device" for the purposes of clarity only and without any intention of being limiting. The recording device is a participant in the IP multimedia session, although preferably the recording device only receives data for recording and/or otherwise monitoring the session. Therefore, the IP multimedia session is preferably a multi-user session, such as a "conference call" for example, even if data is being provided for recording from only one of the participants in the session.

94 Claims, 5 Drawing Sheets



US 7,010,109 B2

Page 2

U.S. PATENT DOCUMENTS

6,006,253 A 12/1999 Kumar et al.
 6,047,060 A 4/2000 Fedorov et al.
 6,108,782 A 8/2000 Fletcher et al.
 6,122,665 A 9/2000 Barr et al.
 6,181,784 B1 1/2001 Duran et al.
 6,288,739 B1 9/2001 Hales et al.
 6,311,231 B1 10/2001 Bateman et al.
 6,320,588 B1 11/2001 Palmer et al.
 6,356,294 B1 3/2002 Martin et al.
 6,418,214 B1 7/2002 Smythe et al.
 6,480,584 B1 11/2002 Duran et al.
 6,490,344 B1 12/2002 Murai et al.
 6,535,909 B1 3/2003 Rust
 6,542,602 B1 4/2003 Elazar
 6,668,273 B1 12/2003 Rust
 6,690,663 B1 2/2004 Culver

FOREIGN PATENT DOCUMENTS

WO WO 02/19620 3/2002

OTHER PUBLICATIONS

vat-LBNL Audio Conferencing Tool, <http://web.archive.org/web/19980126183021/www-nrg.ee.lbl.gov/vat> (5 pages).
 mash-developers@mash.cs.berkeley.edu, "Vic-video conference", <http://web.archive.org/web/19980209092254/mash>. Last updated Mon. Sep. 8, 1997.
 Suchitra Raman & Angela Schuett, Department of EECS, University of California, Berkeley "On-demand Remote Playback", 10 pages.
 International Telecommunication Union, Telecommunication Standardization Sector of ITU, Series H: Audiovisual And Multimedia Systems, Infrastructure of audiovisual services- Systems and terminal equipment for audiovisual services. "Packet-based multimedia communication systems," H.323 (Feb. 1998), 116 pages.
 International Telecommunication Union, Series H: Audiovisual and Multimedia Systems, Infrastructure of audiovisual services-Transmission multiplexing and synchronization, "Call signalling protocols and media stream packetization for packet-based multimedia communication systems," ITU-T Recommendation H.225.0, Printed in Switzerland, Geneva, 1998.
 International Telecommunication Union, Series H: Audiovisual and multimedia Systems, Infrastructure of audiovisual services-Communication procedures, "Control protocol for multimedia communication," Recommendation H.245.
 Intel, Intel support applications & Technical notes, "Intel Internet Video Phone Trial Applet 2.1, The Problems and Pitfalls of Getting H.323 Saftey ThroughFirewalls", http://web.archive.org/web/19980425132417/http://support.intel.com/support/videophone/trial21/h323_wpr.htm#a18 (32 pages).
 Google Groups: comp.security.firewalls "Netmeeting through a packet-filter", http://groups-beta.google.com/group/comp.security.firewalls/browse_thread/thread/c14c3ac7d190a58/a4010ede22ff83a0 (4 pages).
 Google Groups: muc.lists.firewalls, "MS Netmeeting 2.0 abd Raptor Eagle vers. 4.0", http://groups-beta.google.com/group/muc.lists.firewalls/browse_thread/thread/ec0255b64bf36ad4?vc=2 (3 pages).
 Communication Solution CTI News Vendor to Vendor Headlines, <http://www.tmcnet.com/articles/ctmimag/0699/0699news.htm> (15 pages), Jun. 1999.

Radcom, Latency & Loss Measurements, "Breakthrough Internetworking Application for Latency & Loss Measurements from RADCOM," <http://web.archive.org/web/19980527022443/www.radcom-inc.com/press21.htm>, Tel Aviv, Apr. 1997, 2 pages.
 Radcom Press Releases, "New VoIP Testing Applications from RADCOM," Tel Aviv, Feb. 9, 1999, <http://www.radcom.com/radcom/about/pr020999.htm>, 2 pages.
 Radcom, Supported Protocols, "Protocols, protocols, protocols," <http://web.archive.org/web/19980527014033/www.radcom-inc.com/protocol.htm>, 10 pages.
 Radcom, "RADCOM Adds Uni 4.0 Signalling and MPEG-II Support to ATM Analysis Solutions," <http://web.archive.org/web/19980527022611/www.radcom-inc.com/press13.htm>, Tel Aviv, Isreal, Nov., 1996.
 Network Computing, "Voice Over IP, The Way It Should Be", <http://www.nwc.com/1001/1001ws12.html> (5 pages) Jan. 11, 1999.
 Network Computing, "Hear it for yourself: Audio samples from our H.323 tests", <http://www.nwc.com/1001/1001ws2.html>, (7 pages) Jan. 11, 1999.
 Radcom, "Prism200 Multiport WAN/LAN/ATM Analyzer", <http://web.archive.org/web/19980527020144/www.radcom-inc.com/pro-p1.htm> (3 pages).
 Etherpeek, the ag group, inc., "Ethernet network analysis software" user manual for windows Version 2.0 user's guide (171 pages).
 Robert J. Kohlhepp, Network Computing, Interactive Buyer's Guide, Reviews, "AG Groups's Skyline/Satellite Combination Offers Bird's-Eye View of Network Traffic", <http://ibg.networkcomputing.com/612/612skyline.html> (7 pages).
 Mel Beckman, NetMeter 1.0 "See and hear your network", <http://web.archive.org/web/19990224183147/macworld.zdnet.com/pages/june.96/Reviews.2144.html> (3 pages).
 AG Group, Inc. Skyline/Satellite, About Skyline/Satellite "A uick look at Skyline/Satellite", <http://web.archive.org/web/19980206033053/www.aggroup.com/skyline/>.
 Checkpoint, Products & Solutions, Supported Applications "Audio/Video Streaming", <http://web.archive.org/web/19980212233542/www.checkpoint.com/products/technology/index.html>. (6 pages).
 Dameon D. Welch-Albernathy, Check point log Reporter, "Re:[fw1-wizards] tcpdump for Solaris 2.6" <http://oldfaq.phoneboy.com/gurus/200007/msg00081.html> (2 pages).
 Checkpoint Products Solutions, "Stateful Inspection Action," <http://web.archive.org/web/19980212235911/www.checkpoint.com/products/technology/page.2.html>, 4 pages.
 Checkpoint, Products Solutions, Tech note, "Check Point Firewall-1: Extensible Stateful Inspection," <http://web.archive.org/web/19980212235917/www.checkpoint.com/products/technology/page3.html>, 3 pages.
 Radcom, Prismlite: Portable WAN/LAN/ATM Protocol, <http://web.archive.org/web/19980527020156/www.radcom-inc.com/pro-p2.htm>, 3 pages.
 Viewing RTPDump Files, <http://bmrc.berkeley.edu/~davesimp/viewingNotes.html>, David Simpson, Oct. 12, 1996, (1 page).
 S. Waldbusser, Carnegie Mellon University "RFC 1757-Remote Network Monitoring Management Information Base", <http://www.faqs.org.rfcs/rfc1757.html> (65 pages) Feb. 1995.

US 7,010,109 B2

Page 3

-
- 1994 O'Reilly & Associates, Inc. "Microsoft RIFF" <http://netghost.narod.ru/gff/graphics/summary/micriff.htm>, (5 pages).
- Jacobson, Van and Steven McCanne. "vat—LBNL Audio Conferencing Tool." Aug. 1, 1996. Network Research Group, Lawrence Berkeley National Laboratory. (5 pages).
- "Recorder (recorder)." Aug. 27, 1997. MASH Research Group, University of California, Berkeley. (2 pages).
- McCanne, Steven and Van Jacobson. "vic—video conference." MASH Research Group, University of California, Berkeley and Network Research Group, Lawrence Berkeley National Laboratory. Archived as of Feb. 9, 1998. (11 pages).
- "Player (player)." Sep. 8, 1997. MASH Research Group, University of California, Berkeley. (3 pages).
- Simpson, David. "Viewing RTPDump Files." Oct. 12, 1996. University of California, Berkeley. (1 page).
- Walderbusser, S. "RFC 1757—Remote Network Monitoring Management Information Base." Feb. 1995. Network Working Group, Carnegie Mellon University. (65 pages).
- Willis, David. "Voice over IP, The Way It Should Be." Jan. 11, 1999. Network Computing. (5 pages).
- "Hear it for yourself: Audio samples from our H.323 tests." Jan. 11, 1999. Network Computing. (7 pages).
- "PrismLite: Portable WAN/LAN/ATM Protocol Analyzer." Archived as of May 27, 1998. RadCom. (3 pages).
- "CTI News: Vendor To Vendor." Jun. 1999. TMCnet. (15 pages).
- "New VoIP Testing Applications from RADCOM." Feb. 9, 1999. RadCom. (2 pages).
- "Protocol, protocol, protocol . . ." Archived as of May 27, 1998. RadCom. (10 pages).
- "GFF Format Summary: Microsoft RIFF." 1994. O'Reilly & Associates, Inc. (5 pages).
- "Etherpeek for Windows: Version 2.0 User's Guide." 1997. The AG Group Inc. 168 pages.
- "CU-SeeMe." Cornell University. Undated. (2 pages, printed on Oct. 7, 2005).
- "Supported Applications." Archived as of Feb. 12, 1998. Check Point Software Technologies Ltd. (6 pages).
- "Re: [fw1-wizards] tcpdump for Solaris 2.6." Jul. 18, 2000. Firewall Wizards Mailing List. (2 pages).
- "Stateful Inspection Firewall p. 2." Archived as of Feb. 12, 1998. Check Point Software Technologies Ltd. (4 pages).
- "Stateful Inspection Tech Note p. 3." Archived as of Feb. 12, 1998. Check Point Software Technologies Ltd. (3 pages).
- Eldridge, Brett. "MS NetMeeting 2.0 and Raptor Eagles vers. 4.0." May 1, 1997. Google Groups, Google, Inc. (3 pages).
- "Archived Tools Overview." Aug. 30, 1997. MASH Research Group, University of California, Berkeley. (1 page).

U.S. Patent

Mar. 7, 2006

Sheet 1 of 5

US 7,010,109 B2

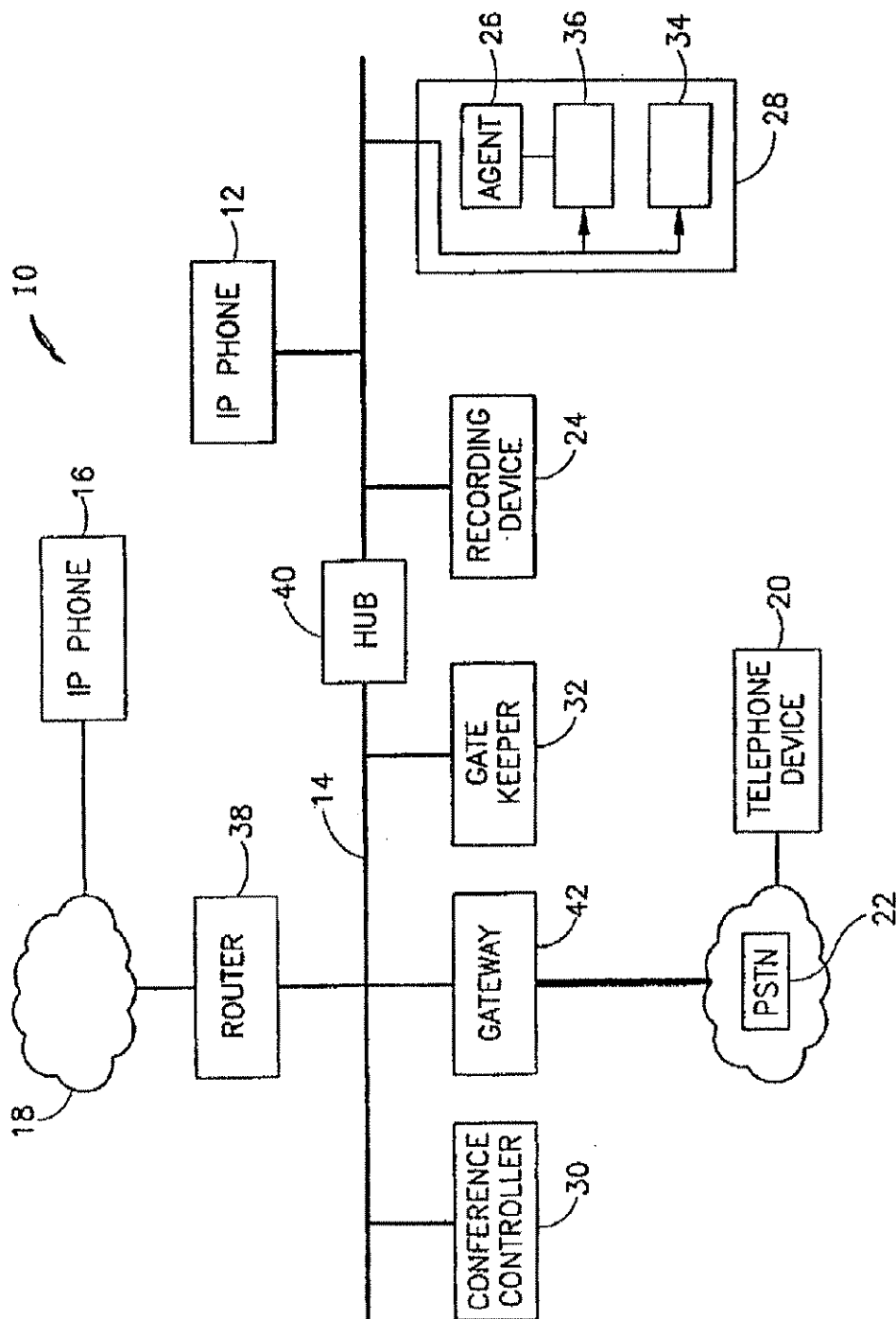


FIG.1

U.S. Patent

Mar. 7, 2006

Sheet 2 of 5

US 7,010,109 B2

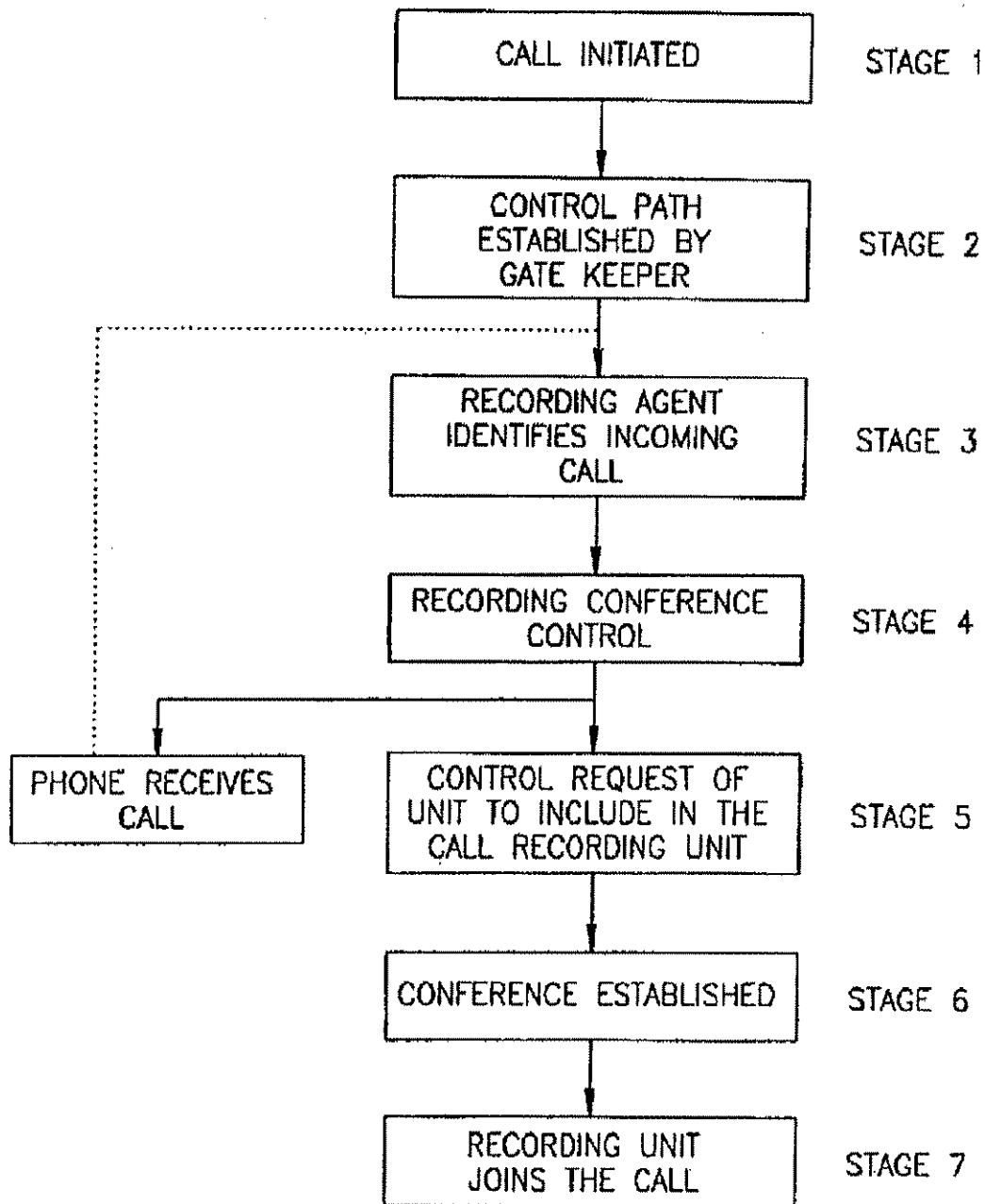


FIG.2

U.S. Patent

Mar. 7, 2006

Sheet 3 of 5

US 7,010,109 B2

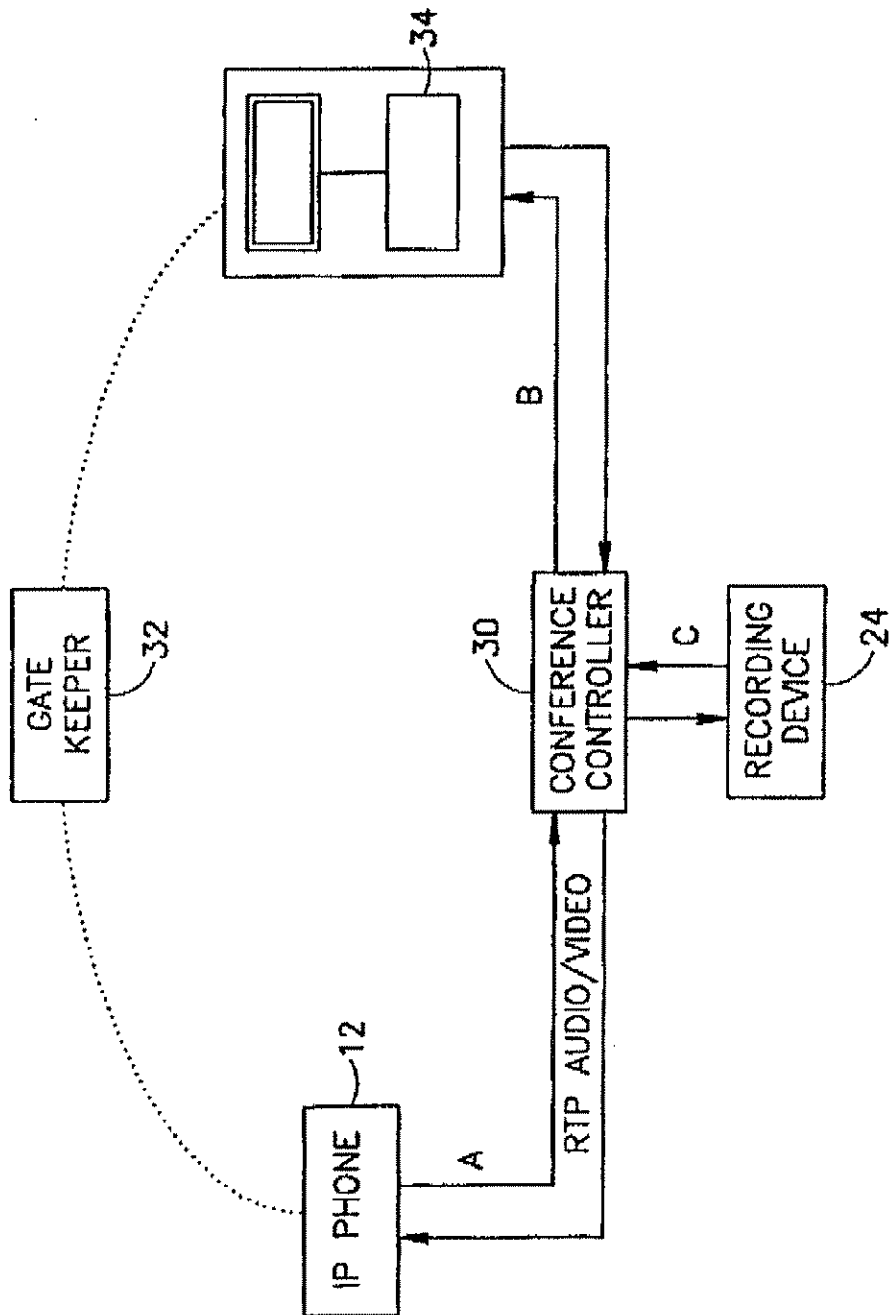


FIG. 3

U.S. Patent

Mar. 7, 2006

Sheet 4 of 5

US 7,010,109 B2

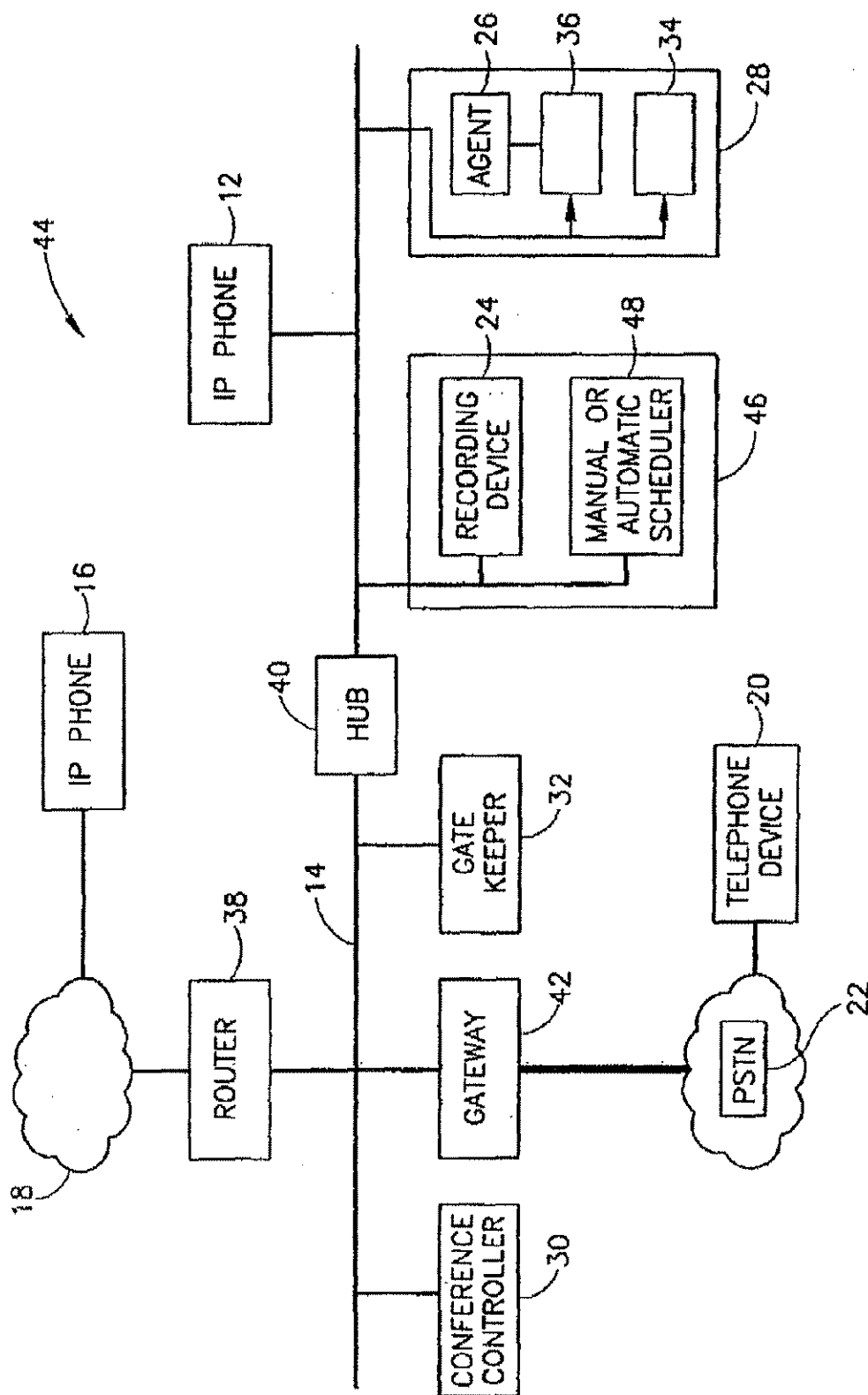


FIG. 4

U.S. Patent

Mar. 7, 2006

Sheet 5 of 5

US 7,010,109 B2

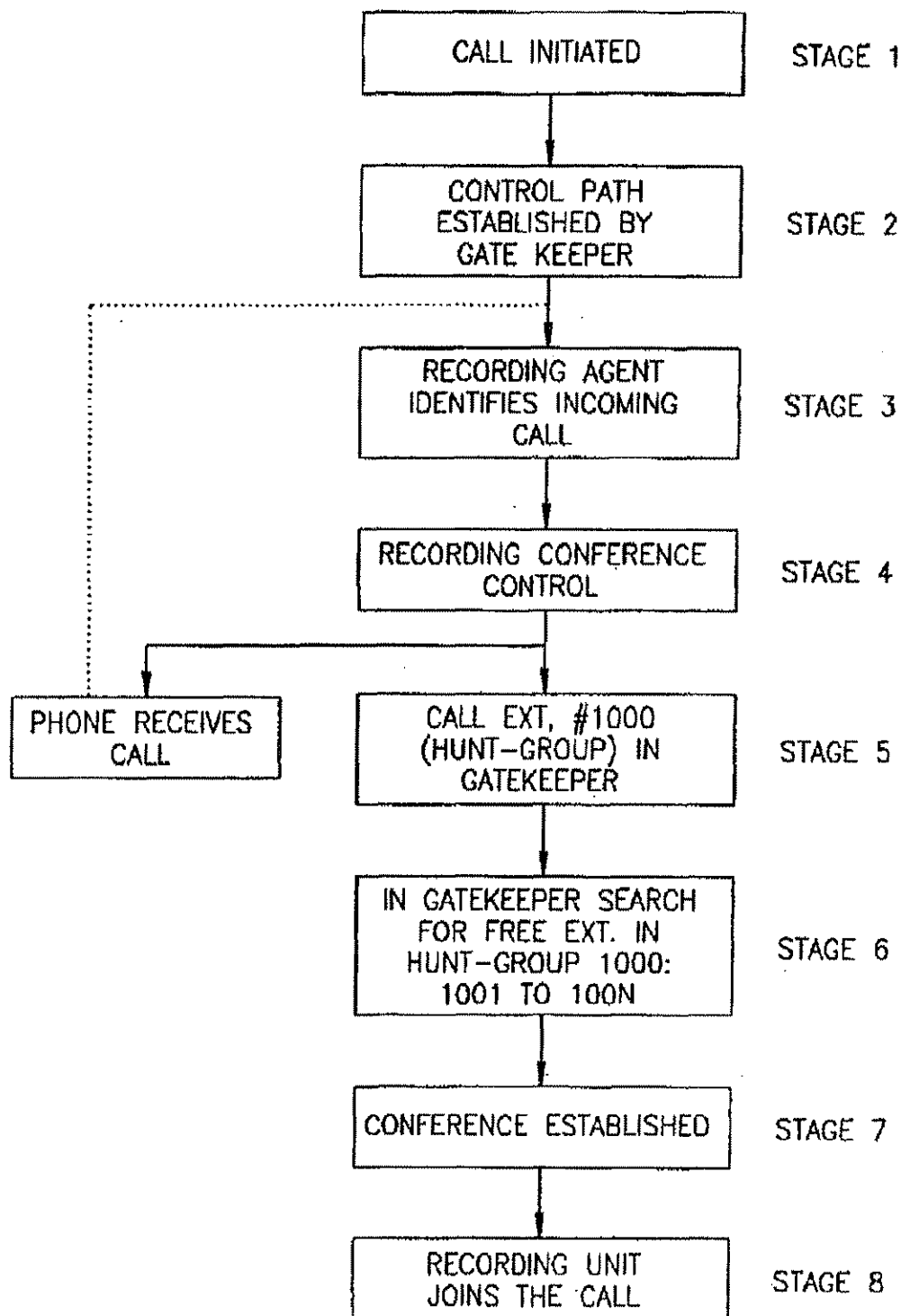


FIG.5

US 7,010,109 B2

1

**DIGITAL RECORDING OF IP BASED
DISTRIBUTED SWITCHING PLATFORM****CROSS REFERENCE TO RELATED
APPLICATIONS**

This application is a continuation of U.S. patent application Ser. No. 10/111,767, filed Jun. 24, 2002, of which is hereby incorporated by specific reference.

FIELD OF THE INVENTION

The present invention relates to a system and a method for recording voice and other data passed through IP multimedia sessions, and in particular, for such a system and method in which recording is triggered with the recording device as a participant in the session.

BACKGROUND OF THE INVENTION

Telecommunication is an important aspect of interactions between individuals, as it enables individuals to communicate without being physically present in the same location, thereby potentially increasing the possibilities for cooperation between such individuals. Simultaneously, an increasing number of telecommunication sessions are being monitored and/or recorded, for example for quality assurance at a "help desk" or other customer support center or service.

Previously, such monitoring or recording was relatively simple in the background art. For example, telephone calls may typically be passed to the individual through a PBX (public exchange) switch or CO (central office), which features a central switching matrix. All telephone calls passing this switch would therefore pass through the central matrix, such that integration of the recording and/or monitoring equipment with the central matrix would enable all such telephone calls to be recorded and/or monitored.

Unfortunately, monitoring and/or recording such telephone calls through the IP multimedia session protocols is not as simple. First, the session is multimedia, such that it may combine two or more different types of data. Second, the session does not pass through a central switching matrix, as IP communication does not feature such a matrix. Thus, such communication is relatively diffuse, even across a WAN (wide area network) or LAN (local area network).

The situation is further complicated by the topology of the IP network, which consists of switch boxes, routers and bridges, and which may prevent any recording and/or monitoring system from accessing such communication sessions that are routed on different network segments. In addition, encrypted sessions add a further element of complexity, as access to such sessions is typically only granted to participants, as only participants have access to the necessary information to decrypt the encrypted session.

SUMMARY OF THE INVENTION

The background art does not teach or suggest a solution to the problem of collecting information about an interactive session over an IP network. The background art also does not teach or suggest a solution to the problem of monitoring and/or recording IP multimedia sessions. In addition, the background art does not teach or suggest a solution to the problem of monitoring and/or recording IP multimedia sessions that are routed on different network segments.

The present invention overcomes these problems of the background art by providing a system and method for

2

recording and/or otherwise monitoring IP multimedia sessions. The present invention features a recording and/or monitoring device, referred to hereinafter as "a recording device" for the purposes of clarity only and without any intention of being limiting. The recording device is a participant in the IP multimedia session, although preferably the recording device only receives data for recording and/or otherwise monitoring the session. Therefore, the IP multimedia session is preferably a multi-user session, such as a "conference call" for example, even if data is being provided for recording from only one of the participants in the session. This implementation of the present invention, as described in greater detail below, overcomes such drawbacks of the background art as the inability to otherwise decrypt encrypted sessions, and recording across network segments.

Hereinafter, the term "separate network portion" includes any separate portion or network across which recording is performed, such as a different network segment and/or network for example.

According to a preferred embodiment of the present invention, the recording device is present on a network with a conference control unit, such as a MCU (multi conference unit) for example. Hereinafter, the term "conference" is used to refer to any multi-participant session, even if only two participants are present, one of which is the device of the present invention. The conference control unit either receives a request to initiate the conference call (multimedia session) from the recording device of the present invention and/or from one of the participating IP communication devices, and/or from some other component on the network. Examples of such communication devices include, but are not limited to, IP telephony devices, "smart" IP telephones and computational devices which include an IP telephony component.

According to another optional but preferred implementation of the present invention, the recording device is the NiceLog.TM. product of Nice Systems Ltd of Ra'anana, Israel.

Hereinafter, the term "computational device" refers to any type of computer hardware system and/or to any type of software operating system, or cellular telephones or any type of hand-held device such as a PDA (personal data assistant), as well as to any type of device having a data processor and/or any type of microprocessor, or any type of device which is capable of performing any function of a computer.

For the present invention, a software application or program could be written in substantially any suitable programming language, which could easily be selected by one of ordinary skill in the art. The programming language chosen should be compatible with the computational device according to which the software application is executed. Examples of suitable programming languages include, but are not limited to, C, C++ and Java.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is herein described, by way of example only, with reference to the accompanying drawings, wherein:

FIG. 1 is a schematic block diagram of an exemplary system according to the present invention;

FIG. 2 is a flowchart of an exemplary method according to the present invention for recording and/or otherwise monitoring IP multimedia sessions;

FIG. 3 is a flow diagram of an optional flow of operations according to the present invention;

US 7,010,109 B2

3

FIG. 4 is a schematic block diagram of a second exemplary system according to the present invention; and

FIG. 5 shows a flowchart of another exemplary method according to the present invention, with regard to the implementation of the present invention with a "hunt group".

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention is of a system and method for recording and/or otherwise monitoring IP multimedia sessions. The present invention features a recording and/or monitoring device, referred to hereinafter as "a recording device" for the purposes of clarity only and without any intention of being limiting. The recording device is a participant in the IP multimedia session, although preferably the recording device only receives data for recording and/or otherwise monitoring the session. Therefore, the IP multimedia session is preferably a multi-user session, such as a "conference call" for example, even if data is being provided for recording from only one of the participants in the session.

Optionally, only a portion of all such multimedia sessions are recorded, although alternatively, all such sessions are recorded. The recording device may optionally receive a command for determining when a session is to be recorded. Alternatively, the recording device may receive data for all such sessions, but may preferably only record certain sessions. More preferably, a scheduler determines whether the session should be recorded, which may optionally be located with the recording device but alternatively is separated on the network.

According to a preferred embodiment of the present invention, the recording device is present on a network with a conference control unit, such as a MCU (multi conference unit) for example. Hereinafter, the term "conference" is used to refer to any multi-participant session, even if only two participants are present, one of which is the device of the present invention. The conference control unit either receives a request to initiate the conference call (multimedia session) from the recording device of the present invention and/or from one of the participating IP communication devices, and/or from some other component on the network. Examples of such communication devices include, but are not limited to, IP telephony devices, "smart" IP telephones and computational devices which include an IP telephony component.

According to another optional but preferred implementation of the present invention, the recording device is the NiceLog.TM. product of Nice Systems Ltd of Ra'anana, Israel.

According to other optional but preferred embodiments of the present invention, the IP multimedia session may also include one or more non-IP telephony devices, such as a telephone device communicating through the PSTN (public switched telephony network). For this embodiment, the system of the present invention preferably features a gateway for receiving such communication and for enabling the data to be passed to other components of the present invention, including but not limited to the recording device.

According to another optional but preferred implementation of the present invention, the system and method of the present invention are enabled for "hunt groups", which use a plurality of virtual telephone numbers rather than fixed telephone lines that are reserved for particular telephone numbers. Hunt groups are well known in the art; one example of a suitable reference is found in "Newton's

4

Telecom Dictionary", 16th Expanded & Updated Edition, by Harry Newton (published in 2000, by Telecom Books; page 414), which is incorporated by reference as if fully set forth herein. Hereinafter, the term "hunt group" refers to any type of virtual or non-fixed telephone extension systems, in which a central control unit of some type, such as the gatekeeper of the present invention, determines the physical extension which is used.

The present invention may also optionally be implemented with a number of well known protocols in the background art for multimedia IP sessions, including but not limited to H.323, RTP (real time protocol), RTCP (real time control protocol), H.225 and H.245; as well as CODECs for encoding/decoding the multimedia data for such sessions, including but not limited to, G.711, G.723, G.722, G.728, H.261 and H.263; all of which are hereby incorporated by reference as if fully set forth herein. In addition, references may be found at <http://www.normos.org/ietf/rfc/rfc18-89.txt> as of Aug. 17, 2001, which are also hereby incorporated by reference as if fully set forth herein, including all links and other data/Web pages found at the Web site. Further information may also be found in U.S. Pat. No. 6,122,665, issued Sep. 19, 2000, which is also incorporated by reference as if fully set forth herein.

The principles and operation of the method according to the present invention may be better understood with reference to the drawings and the accompanying description. It should be noted that the present invention is described with regard to IP telephony for the purposes of clarity only and without any intention of being limiting.

Referring now to the drawings, FIG. 1 shows an illustrative system 10 for recording and/or otherwise monitoring an IP communication session, which may optionally be a multimedia session. The session may optionally be initiated at any one of an IP telephone 12 on a LAN (local area network) 14; an IP telephone 16 on a WAN (wide area network) 18; and a telephony device 20 communicating through a PSTN (public switched telephony network) 22. Examples of suitable IP telephones include but are not limited to, VIP 30 or SP+12 (Cisco Inc., San Jose, Calif., USA). Preferably, the actual handling of the session is slightly different for each of these different initiating devices, as described in greater detail below.

As shown, LAN 14 features a recording device 24. According to another optional but preferred implementation of the present invention, recording device 24 is the NiceLog.TM. product of Nice Systems Ltd of Ra'anana, Israel. This product features a monitor for monitoring activity through voice telephony on an IP network. Although the activity is monitored through voice telephony protocols, other types of data may also optionally be monitored, such as video and audio data transmissions. The monitor component of the NiceLog.TM. product includes a recording function to record these voice and other types of data transmissions. For example, the recording function may be manually activated to start recording. Further details may be found in the User's Manual of the NiceLog.TM. product.

Recording device 24 is preferably in communication with a recording agent 26 for controlling the process of recording, although optionally both recording device 24 and recording agent 26 may be present in a single device, although separate devices are preferred. Alternatively, recording device 24 may optionally perform all of these functions. Recording agent 26 is preferably operated as a software module by a computational device 28. According to the present invention, upon initiation of the IP multimedia session, recording agent 26 determines that the session has been initiated and

US 7,010,109 B2

5

directs recording device 24 to record the session. Optionally, only certain sessions are recorded. In order to support recording, the multimedia session is constructed as a conference call, and recording device 24 then becomes a participant in that conference call.

FIG. 1 shows one exemplary implementation for supporting these functions. As shown, LAN 14 also optionally and preferably is connected to a conference controller 30, such as an MCU for example. Conference controller 30 establishes the conference call. Preferably, LAN 14 connects to a gatekeeper 32 according to the H.323 protocol, which translates telephone numbers to IP addresses, and therefore enables the initiating device to locate the other communication device (if present on LAN 14). A non-limiting example of gatekeeper 32 is the MCS 7820 product (Cisco Inc., San Jose, Calif., USA). Gatekeeper 32 may optionally be assisted in performing IP address resolution by a DHCP server (not shown), which is connected to LAN 14. DHCP server assigns IP addresses to IP telephone 12 and to other IP telephones and devices; the assigned addresses are then passed to gatekeeper 32 for performing IP address resolution.

For the first example of initiating device previously given, IP telephone 12 on LAN 14 initiates the session, as explained also with regard to the flowchart of FIG. 2, showing an exemplary method according to the present invention for recording and/or otherwise monitoring IP multimedia sessions. For example, IP telephone 12 may contact gatekeeper 32 to initiate the session with an IP telephone 34 on computational device 28 in stage 1. Both participants are therefore connected through LAN 14.

In stage 2, the control path is established by gatekeeper 32, for example according to the H.323 protocol, in order for the IP session to be initiated. In stage 3, if recording device 24 is not present and/or operational, preferably the normal IP communication session is enabled with IP telephone 34. Alternatively, if recording device 24 is present, then recording agent 26 preferably identifies the incoming request to initiate the session.

In stage 4, a recording agent control module 36, shown with regard to FIG. 1, preferably controls the conference call recording. Optionally and more preferably, recording agent control module 36 sends a request to initiate the conference call to gatekeeper 32. This request preferably includes a request to include recording device 24 in the conference call.

In stage 5, gatekeeper 32 sends a request to conference controller 30 to initiate the IP multimedia session, with recording device 24 as a participant thereof. In stage 6, conference controller 30 initiates the conference call between IP telephone 12 and IP telephone 34. In stage 7, recording device 24 is preferably added to the conference call.

A similar operation is performed if the session is to be established with IP telephone 16 on WAN 18. As shown in FIG. 1, WAN 18 is optionally connected to LAN 14 through a router 38 (LAN 14 may optionally feature a hub 40). IP telephone 12 may again initiate the session by contacting gatekeeper 32; the remaining stages are performed substantially as previously described. Alternatively, IP telephone 16 may initiate the session. In order for IP telephone 16 to initiate the session and the recording, preferably IP telephone 16 features recording agent 26 and recording agent control module 36 as part of a single device. It should be noted that only one of IP telephone 12 and IP telephone 16 requires recording agent 26 and recording agent control module 36, operated directly by the IP telephone itself (in the case of a "smart telephone"), or alternatively operated by

6

a computational device which also operates the IP telephone, in order for the session to be recorded.

The operation is preferably adjusted somewhat if a telephony device 20 communicating through a PSTN 22 is contacted by IP telephone 12 to initiate the multimedia call and/or if telephony device 20 initiates the call. In both cases, communication to and from telephony device 20 passes through a gateway 42, for example in order to translate regular PSTN 22 communication to IP-based communication, such as H.323 protocol-based communication for example. Gateway 42 then preferably contacts gatekeeper 32 in order for telephony device 20 to be recognized as a participant in the session. The remaining functions are similar to those shown in FIG. 2. Gateway 42 may optionally be implemented as a Cisco Internet Router 3620, for example (Cisco Inc., San Jose, Calif., USA).

FIG. 3 shows a flow diagram of an optional flow of operations according to the present invention. As shown, IP telephone 12 initiates the session, through gatekeeper 32, to IP telephone 34. The session is implemented as a conference call. Conference controller 30 enables recording device 24 to participate in the conference call, as well as preferably enabling the conference call itself. It should be noted that typically that only information passing through arrows "A" and "B", from each of IP telephone 12 and IP telephone 34 respectively, is recorded. Also, optionally and preferably, recording device 24 only receives communication through arrow "C". For this implementation, recording device 24 preferably has at least one, and more preferably a plurality of, reserved telephone numbers which correspond to actual telephone lines. Video and/or audio data may optionally be captured according to the RTP (real time protocol) protocol.

FIG. 4 shows another exemplary system 44 according to the present invention for selective recording of sessions. Similar components to FIG. 1 retain the same numbering. Now, recording device 24 is preferably contained within a selective recorder 46, which also features a scheduler 48. Scheduler 48 may optionally be manual or automatic. For the latter implementation, scheduler 48 may optionally analyze information about the IP multimedia session, such as the identity of the initiating and/or receiving device, in order to determine whether the session should be recorded. For the manual implementation, the user at the receiving and/or initiating IP telephony device may optionally determine whether the session should be recorded.

FIG. 5 shows a flowchart of another exemplary method according to the present invention, with regard to the implementation of the present invention with a "hunt group". As previously described, hunt groups use a plurality of virtual telephone numbers rather than fixed telephone lines that are reserved for particular telephone numbers. The present invention supports recording and/or otherwise monitoring IP multimedia sessions with such hunt groups as shown in FIG. 5.

This preferred method is similar to that of FIG. 2 for stages 1-4. In stage 5, however, the gatekeeper identifies the hunt group which has been called.

In stage 6, the gatekeeper searches for a free telephone line within that particular hunt group. In stage 7, the conference call is established through the conference controller, and the recording device joins the conference call in stage 8, as previously described.

According to optional but preferred implementations of the present invention, any of the above embodiments may be optionally implemented with a "smart" telephone device in place of the computational device for operating the recording agent and/or the recording agent control module 36.

US 7,010,109 B2

7

While the invention has been described with respect to a limited number of embodiments, it will be appreciated that many variations, modifications and other applications of the invention may be made.

What is claimed is:

1. A method for recording at least a portion of one or more of a plurality of IP data sessions, each being between at least a first communication device and a second communication device through a network by a recording device, comprising for each IP data session:

initiating the data session by said first communication device with said second communication device;

implementing the data session as a conference call through a conference controller such that said first and second communication devices are connected, respectively, as first and second participants;

using the conference controller, selectively entering the recording device to said conference call as an additional participant, wherein the recording device is distinct from the first and second communication devices yet receives as the additional participant at least the portion of the IP data session from each of the first and second participants; and

recording at least the portion of the IP data session received as the additional participant of said conference call using said recording device.

2. The method of claim 1, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

3. The method of claim 1, including the additional step of permitting a user of at least one of the first and second communication devices to determine whether the session is to be recorded prior to entering the recording device as the additional participant.

4. The method of claim 1, wherein the connection of the second communication device is established by the conference controller by:

passing telephone numbers to a gatekeeper for performing IP address resolution, and

using a resolved IP address of the second communication device for connecting the second communication device to the conference call.

5. The method of claim 1, wherein the step of selectively entering the recording device to said conference call is in response to a command that the data session is to be recorded.

6. The method of claim 5, including the additional step of providing the command from a scheduler.

7. The method of claim 6, including the additional step of locating the scheduler with the recording device.

8. The method of claim 6, including the additional step of analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

9. The method of claim 8, wherein the information includes the identity of at least one of the first and second communication devices.

10. The method of claim 1, wherein the IP data session is either an IP telephony session or an IP multimedia session.

11. The method of claim 1, wherein the step of initiating the data session is detected by a recording agent, and wherein said recording agent contacts the recording device.

12. The method of claim 1, wherein said conference controller is a MCU.

8

13. The method of claim 1, wherein the conference controller implements said conference call in response to a request to initiate the conference call.

14. The method of claim 13, wherein the request is from at least one of the recording device, the first communication device, the second communication device, and an other component on the network.

15. The method of claim 1, wherein said first communication device is a gateway for receiving communication through a PSTN.

16. The method of claim 1, wherein the recording device joins the data session performed through a hunt group.

17. The method of claim 16, including the additional step of identifying the hunt group using a gatekeeper.

18. The method of claim 1, wherein at least one of the first communication device and the second communication device is a non-IP telephony device.

19. The method of claim 18, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

20. The method of claim 18, wherein the connection of the second communication device is established by the conference controller by:

passing telephone numbers to a gatekeeper for performing IP address resolution, and

using a resolved IP address of the second communication device for connecting the second communication device to the conference call.

21. The method of claim 18, wherein the step of selectively entering the recording device to said conference call is in response to a command that the data session is to be recorded.

22. The method of claim 21, including the additional step of providing the command from a scheduler.

23. The method of claim 22, including the additional step of locating the scheduler with the recording device.

24. The method of claim 22, including the additional step of analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

25. The method of claim 24, wherein the information includes the identity of at least one of the first and second communication devices.

26. The method of claim 18, wherein the step of initiating the data session is detected by a recording agent, and wherein said recording agent contacts the recording device.

27. The method of claim 26, wherein said conference controller is a MCU.

28. The method of claim 26, wherein the conference controller implements said conference call in response to a request to initiate the conference call.

29. The method of claim 18, wherein the recording device joins the data session performed through a hunt group.

30. The method of claim 29, including the additional step of identifying the hunt group using a gatekeeper.

31. The method of claim 1, including the additional steps of passing telephone numbers to a gatekeeper for performing IP address resolution and using a resolved IP address of the second communication device in connecting the second communication device to the conference call, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

US 7,010,109 B2

9

32. The method of claim 31, wherein the recording device is directed to enter said conference call in response to a command that the data session is to be recorded.

33. The method of claim 32, including the additional steps of:

providing the command from a scheduler; and
analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

34. The method of claim 33, wherein the information includes the identity of at least one of the first and second communication devices.

35. The method of claim 1, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant in response to a command that the data session is to be recorded.

36. The method of claim 35, including the additional steps of:

providing the command from a scheduler; and
analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

37. The method of claim 36, wherein the information includes the identity of at least one of the first and second communication devices.

38. The method of claim 1, including the additional steps of:

detecting the step of initiating the data session using a recording agent,
contacting the recording device using the recording agent, and
receiving a request to initiate the conference call and performing the implementing step in response to the request,
wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

39. The method of claim 38, including the additional steps of passing telephone numbers to a gatekeeper for performing IP address resolution and using a resolved IP address of the second communication device in connecting the second communication device to the conference call.

40. The method of claim 38, wherein the recording device is directed to enter said conference call in response to a command that the data session is to be recorded.

41. The method of claim 38, including the additional steps of:

providing the command from a scheduler; and
analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

42. The method of claim 41, wherein the information includes the identity of at least one of the first and second communication devices.

43. The method of claim 1, wherein the step of selectively entering the recording device to said conference call includes the steps of:

identifying a hunt group using a gatekeeper;
directing the recording device to enter said conference call as the additional participant in response to a command that the data session is to be recorded; and
joining the recording device to the data session through the hunt group.

10

44. The method of claim 43, including the additional steps of passing telephone numbers to a gatekeeper for performing IP address resolution and using a resolved IP address of the second communication device in connecting the second communication device to the conference call.

45. The method of claim 43, wherein the recording device is directed to enter said conference call in response to a command that the data session is to be recorded.

46. The method of claim 45, including the additional steps of:

providing the command from a scheduler; and
analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

47. The method of claim 46, wherein the information includes the identity of at least one of the first and second communication devices.

48. A method for recording at least a portion of an IP data session between at least a first communication device and a second communication device through a network by a recording device, comprising:

initiating the data session by said first communication device with said second communication device;
implementing the data session as a conference call through a conference controller such that said first and second communication devices are connected, respectively, as first and second participants;
using the conference controller, selectively entering the recording device to said conference call as an additional participant, wherein the recording device is distinct from the first and second communication devices yet receives as the additional participant at least the portion of the IP data session from each of the first and second participants; and

recording at least the portion of the IP data session received as the additional participant of said conference call using said recording device.

49. The method of claim 48, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

50. The method of claim 48, including the additional step of permitting a user of at least one of the first and second communication devices to determine whether the session is to be recorded prior to entering the recording device as the additional participant.

51. The method of claim 48, wherein the connection of the second communication device is established by the conference controller by:

passing telephone numbers to a gatekeeper for performing IP address resolution, and
using a resolved IP address of the second communication device for connecting the second communication device to the conference call.

52. The method of claim 48, wherein the step of selectively entering the recording device to said conference call is in response to a command that the data session is to be recorded.

53. The method of claim 52, including the additional step of providing the command from a scheduler.

54. The method of claim 53, including the additional step of locating the scheduler with the recording device.

55. The method of claim 53, including the additional step of analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

US 7,010,109 B2

11

56. The method of claim 55, wherein the information includes the identity of at least one of the first and second communication devices.

57. The method of claim 48, wherein the IP data session is either an IP telephony session or an IP multimedia session.

58. The method of claim 48, wherein the step of initiating the data session is detected by a recording agent, and wherein said recording agent contacts the recording device.

59. The method of claim 48, wherein said conference controller is a MCU.

60. The method of claim 48, wherein the conference controller implements said conference call in response to a request to initiate the conference call.

61. The method of claim 60, wherein the request is from at least one of the recording device, the first communication device, the second communication device, and an other component on the network.

62. The method of claim 48, wherein said first communication device is a gateway for receiving communication through a PSTN.

63. The method of claim 48, wherein the recording device joins the data session performed through a hunt group.

64. The method of claim 63, including the additional step of identifying the hunt group using a gatekeeper.

65. The method of claim 48, wherein at least one of the first communication device and the second communication device is a non-IP telephony device.

66. The method of claim 65, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

67. The method of claim 65, wherein the connection of the second communication device is established by the conference controller by:

passing telephone numbers to a gatekeeper for performing IP address resolution, and

using a resolved IP address of the second communication device for connecting the second communication device to the conference call.

68. The method of claim 65, wherein the step of selectively entering the recording device to said conference call is in response to a command that the data session is to be recorded.

69. The method of claim 68, including the additional step of providing the command from a scheduler.

70. The method of claim 69, including the additional step of locating the scheduler with the recording device.

71. The method of claim 69, including the additional step of analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

72. The method of claim 71, wherein the information includes the identity of at least one of the first and second communication devices.

73. The method of claim 65 wherein the step of initiating the data session is detected by a recording agent, and wherein said recording agent contacts the recording device.

74. The method of claim 73, wherein said conference controller is a MCU.

75. The method of claim 73, wherein the conference controller implements said conference call in response to a request to initiate the conference call.

76. The method of claim 65, wherein the recording device joins the data session performed through a hunt group.

77. The method of claim 76, including the additional step of identifying the hunt group using a gatekeeper.

12

78. The method of claim 48, including the additional steps of passing telephone numbers to a gatekeeper for performing IP address resolution and using a resolved IP address of the second communication device in connecting the second communication device to the conference call, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

79. The method of claim 78, wherein the recording device is directed to enter said conference call in response to a command that the data session is to be recorded.

80. The method of claim 79, including the additional steps of:

providing the command from a scheduler; and
analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

81. The method of claim 80, wherein the information includes the identity of at least one of the first and second communication devices.

82. The method of claim 48, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant in response to a command that the data session is to be recorded.

83. The method of claim 82, including the additional steps of:

providing the command from a scheduler; and
analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

84. The method of claim 83, wherein the information includes the identity of at least one of the first and second communication devices.

85. The method of claim 48, including the additional steps of:

detecting the step of initiating the data session using a recording agent,
contacting the recording device using the recording agent, and

receiving a request to initiate the conference call and performing the implementing step in response to the request, wherein the step of selectively entering the recording device to said conference call includes the step of directing the recording device to enter said conference call as the additional participant when a data session has been initiated.

86. The method of claim 85, including the additional steps of passing telephone numbers to a gatekeeper for performing IP address resolution and using a resolved IP address of the second communication device in connecting the second communication device to the conference call.

87. The method of claim 85, wherein the recording device is directed to enter said conference call in response to a command that the data session is to be recorded.

88. The method of claim 85, including the additional steps of:

providing the command from a scheduler; and
analyzing information about the IP data session at the scheduler to determine whether the IP data session is to be recorded.

89. The method of claim 88, wherein the information includes the identity of at least one of the first and second communication devices.

US 7,010,109 B2

13

90. The method of claim 48, wherein the step of selectively entering the recording device to said conference call includes the steps of:

identifying a hunt group using a gatekeeper;
directing the recording device to enter said conference 5
call as the additional participant in response to a
command that the data session is to be recorded; and
joining the recording device to the data session through
the hunt group.

91. The method of claim 90, including the additional steps 10
of passing telephone numbers to a gatekeeper for performing
IP address resolution and using a resolved IP address of the
second communication device in connecting the second
communication device to the conference call.

14

92. The method of claim 90, wherein the recording device
is directed to enter said conference call in response to a
command that the data session is to be recorded.

93. The method of claim 92, including the additional steps
of:

providing the command from a scheduler; and
analyzing information about the IP data session at the
scheduler to determine whether the IP data session is to
be recorded.

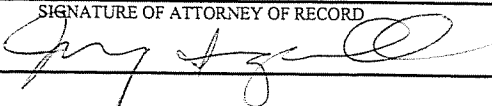
94. The method of claim 93, wherein the information
includes the identity of at least one of the first and second
communication devices.

* * * * *

JS 44 (Rev. 11/04)

CIVIL COVER SHEET

The JS 44 civil cover sheet and the information contained herein neither replace nor supplement the filing and service of pleadings or other papers as required by law, except as provided by local rules of court. This form, approved by the Judicial Conference of the United States in September 1974, is required for the use of the Clerk of Court for the purpose of initiating the civil docket sheet. (SEE INSTRUCTIONS ON THE REVERSE OF THE FORM.)

I. (a) PLAINTIFFS NICE SYSTEMS, INC. and NICE SYSTEMS LTD. (b) County of Residence of First Listed Plaintiff <u>Bergen</u> (EXCEPT IN U.S. PLAINTIFF CASES) (c) Attorney's (Firm Name, Address, and Telephone Number) Josy W. Ingersoll (No. 1088) Young Conaway Stargatt & Taylor, LLP, 1000 West Street, 17th Floor, The Broadwing Building, Wilmington, DE 19800-0204			DEFENDANTS WITNESS SYSTEMS, INC. County of Residence of First Listed Defendant _____ (IN U.S. PLAINTIFF CASES ONLY) NOTE: IN LAND CONDEMNATION CASES, USE THE LOCATION OF THE LAND INVOLVED. Attorneys (If Known) _____													
II. BASIS OF JURISDICTION (Place an "X" in One Box Only)			III. CITIZENSHIP OF PRINCIPAL PARTIES (Place an "X" in One Box for Plaintiff and One Box for Defendant)													
<input type="checkbox"/> 1 U.S. Government Plaintiff <input type="checkbox"/> 2 U.S. Government Defendant			<input checked="" type="checkbox"/> 3 Federal Question (U.S. Government Not a Party) <input type="checkbox"/> 4 Diversity (Indicate Citizenship of Parties in Item III)													
IV. NATURE OF SUIT (Place an "X" in One Box Only)			<table border="1" style="width:100%; border-collapse: collapse;"> <tr> <th style="text-align: left;">CONTRACT</th> <th style="text-align: left;">TORTS</th> <th style="text-align: left;">FORFEITURE/PENALTY</th> <th style="text-align: left;">BANKRUPTCY</th> <th style="text-align: left;">OTHER STATUTES</th> </tr> <tr> <td style="vertical-align: top;"> <input type="checkbox"/> 110 Insurance <input type="checkbox"/> 120 Marine <input type="checkbox"/> 130 Miller Act <input type="checkbox"/> 140 Negotiable Instrument <input type="checkbox"/> 150 Recovery of Overpayment & Enforcement of Judgment <input type="checkbox"/> 151 Medicare Act <input type="checkbox"/> 152 Recovery of Defaulted Student Loans (Excl. Veterans) <input type="checkbox"/> 153 Recovery of Overpayment of Veteran's Benefits <input type="checkbox"/> 160 Stockholders' Suits <input type="checkbox"/> 190 Other Contract <input type="checkbox"/> 195 Contract Product Liability <input type="checkbox"/> 196 Franchise </td> <td style="vertical-align: top;"> PERSONAL INJURY <input type="checkbox"/> 310 Airplane <input type="checkbox"/> 315 Airplane Product Liability <input type="checkbox"/> 320 Assault, Libel & Slander <input type="checkbox"/> 330 Federal Employers' Liability <input type="checkbox"/> 340 Marine <input type="checkbox"/> 345 Marine Product Liability <input type="checkbox"/> 350 Motor Vehicle <input type="checkbox"/> 355 Motor Vehicle Product Liability <input type="checkbox"/> 360 Other Personal Injury CIVIL RIGHTS <input type="checkbox"/> 441 Voting <input type="checkbox"/> 442 Employment <input type="checkbox"/> 443 Housing/Accommodations <input type="checkbox"/> 444 Welfare <input type="checkbox"/> 445 Amer. w/Disabilities - Employment <input type="checkbox"/> 446 Amer. w/Disabilities - Other <input type="checkbox"/> 440 Other Civil Rights </td> <td style="vertical-align: top;"> PERSONAL INJURY <input type="checkbox"/> 362 Personal Injury - Med. Malpractice <input type="checkbox"/> 365 Personal Injury - Product Liability <input type="checkbox"/> 368 Asbestos Personal Injury Product Liability PERSONAL PROPERTY <input type="checkbox"/> 370 Other Fraud <input type="checkbox"/> 371 Truth in Lending <input type="checkbox"/> 380 Other Personal Property Damage <input type="checkbox"/> 385 Property Damage Product Liability PRISONER PETITIONS <input type="checkbox"/> 510 Motions to Vacate Sentence <input type="checkbox"/> Habeas Corpus: <input type="checkbox"/> 530 General <input type="checkbox"/> 535 Death Penalty <input type="checkbox"/> 540 Mandamus & Other <input type="checkbox"/> 550 Civil Rights <input type="checkbox"/> 555 Prison Condition </td> <td style="vertical-align: top;"> <input type="checkbox"/> 610 Agriculture <input type="checkbox"/> 620 Other Food & Drug <input type="checkbox"/> 625 Drug Related Seizure of Property 21 USC 881 <input type="checkbox"/> 630 Liquor Laws <input type="checkbox"/> 640 R.R. & Truck <input type="checkbox"/> 650 Airline Regs. <input type="checkbox"/> 660 Occupational Safety/Health <input type="checkbox"/> 690 Other LABOR <input type="checkbox"/> 710 Fair Labor Standards Act <input type="checkbox"/> 720 Labor/Mgmt. Relations <input type="checkbox"/> 730 Labor/Mgmt. Reporting & Disclosure Act <input type="checkbox"/> 740 Railway Labor Act <input type="checkbox"/> 790 Other Labor Litigation <input type="checkbox"/> 791 Empl. Ret. Inc. Security Act </td> <td style="vertical-align: top;"> <input type="checkbox"/> 422 Appeal 28 USC 158 <input type="checkbox"/> 423 Withdrawal 28 USC 157 PROPERTY RIGHTS <input type="checkbox"/> 820 Copyrights <input checked="" type="checkbox"/> 830 Patent <input type="checkbox"/> 840 Trademark SOCIAL SECURITY <input type="checkbox"/> 861 HIA (1395f) <input type="checkbox"/> 862 Black Lung (923) <input type="checkbox"/> 863 DIWC/DIWW (405(g)) <input type="checkbox"/> 864 SSID Title XVI <input type="checkbox"/> 865 RSI (405(g)) FEDERAL TAX SUITS <input type="checkbox"/> 870 Taxes (U.S. Plaintiff or Defendant) <input type="checkbox"/> 871 IRS—Third Party 26 USC 7609 </td> <td style="vertical-align: top;"> <input type="checkbox"/> 400 State Reapportionment <input type="checkbox"/> 410 Antitrust <input type="checkbox"/> 430 Banks and Banking <input type="checkbox"/> 450 Commerce <input type="checkbox"/> 460 Deportation <input type="checkbox"/> 470 Racketeer Influenced and Corrupt Organizations <input type="checkbox"/> 480 Consumer Credit <input type="checkbox"/> 490 Cable/Sat TV <input type="checkbox"/> 810 Selective Service <input type="checkbox"/> 850 Securities/Commodities/Exchange <input type="checkbox"/> 875 Customer Challenge 12 USC 3410 <input type="checkbox"/> 890 Other Statutory Actions <input type="checkbox"/> 891 Agricultural Acts <input type="checkbox"/> 892 Economic Stabilization Act <input type="checkbox"/> 893 Environmental Matters <input type="checkbox"/> 894 Energy Allocation Act <input type="checkbox"/> 895 Freedom of Information Act <input type="checkbox"/> 900 Appeal of Fee Determination Under Equal Access to Justice <input type="checkbox"/> 950 Constitutionality of State Statutes </td> </tr> </table>			CONTRACT	TORTS	FORFEITURE/PENALTY	BANKRUPTCY	OTHER STATUTES	<input type="checkbox"/> 110 Insurance <input type="checkbox"/> 120 Marine <input type="checkbox"/> 130 Miller Act <input type="checkbox"/> 140 Negotiable Instrument <input type="checkbox"/> 150 Recovery of Overpayment & Enforcement of Judgment <input type="checkbox"/> 151 Medicare Act <input type="checkbox"/> 152 Recovery of Defaulted Student Loans (Excl. Veterans) <input type="checkbox"/> 153 Recovery of Overpayment of Veteran's Benefits <input type="checkbox"/> 160 Stockholders' Suits <input type="checkbox"/> 190 Other Contract <input type="checkbox"/> 195 Contract Product Liability <input type="checkbox"/> 196 Franchise	PERSONAL INJURY <input type="checkbox"/> 310 Airplane <input type="checkbox"/> 315 Airplane Product Liability <input type="checkbox"/> 320 Assault, Libel & Slander <input type="checkbox"/> 330 Federal Employers' Liability <input type="checkbox"/> 340 Marine <input type="checkbox"/> 345 Marine Product Liability <input type="checkbox"/> 350 Motor Vehicle <input type="checkbox"/> 355 Motor Vehicle Product Liability <input type="checkbox"/> 360 Other Personal Injury CIVIL RIGHTS <input type="checkbox"/> 441 Voting <input type="checkbox"/> 442 Employment <input type="checkbox"/> 443 Housing/Accommodations <input type="checkbox"/> 444 Welfare <input type="checkbox"/> 445 Amer. w/Disabilities - Employment <input type="checkbox"/> 446 Amer. w/Disabilities - Other <input type="checkbox"/> 440 Other Civil Rights	PERSONAL INJURY <input type="checkbox"/> 362 Personal Injury - Med. Malpractice <input type="checkbox"/> 365 Personal Injury - Product Liability <input type="checkbox"/> 368 Asbestos Personal Injury Product Liability PERSONAL PROPERTY <input type="checkbox"/> 370 Other Fraud <input type="checkbox"/> 371 Truth in Lending <input type="checkbox"/> 380 Other Personal Property Damage <input type="checkbox"/> 385 Property Damage Product Liability PRISONER PETITIONS <input type="checkbox"/> 510 Motions to Vacate Sentence <input type="checkbox"/> Habeas Corpus: <input type="checkbox"/> 530 General <input type="checkbox"/> 535 Death Penalty <input type="checkbox"/> 540 Mandamus & Other <input type="checkbox"/> 550 Civil Rights <input type="checkbox"/> 555 Prison Condition	<input type="checkbox"/> 610 Agriculture <input type="checkbox"/> 620 Other Food & Drug <input type="checkbox"/> 625 Drug Related Seizure of Property 21 USC 881 <input type="checkbox"/> 630 Liquor Laws <input type="checkbox"/> 640 R.R. & Truck <input type="checkbox"/> 650 Airline Regs. <input type="checkbox"/> 660 Occupational Safety/Health <input type="checkbox"/> 690 Other LABOR <input type="checkbox"/> 710 Fair Labor Standards Act <input type="checkbox"/> 720 Labor/Mgmt. Relations <input type="checkbox"/> 730 Labor/Mgmt. Reporting & Disclosure Act <input type="checkbox"/> 740 Railway Labor Act <input type="checkbox"/> 790 Other Labor Litigation <input type="checkbox"/> 791 Empl. Ret. Inc. Security Act	<input type="checkbox"/> 422 Appeal 28 USC 158 <input type="checkbox"/> 423 Withdrawal 28 USC 157 PROPERTY RIGHTS <input type="checkbox"/> 820 Copyrights <input checked="" type="checkbox"/> 830 Patent <input type="checkbox"/> 840 Trademark SOCIAL SECURITY <input type="checkbox"/> 861 HIA (1395f) <input type="checkbox"/> 862 Black Lung (923) <input type="checkbox"/> 863 DIWC/DIWW (405(g)) <input type="checkbox"/> 864 SSID Title XVI <input type="checkbox"/> 865 RSI (405(g)) FEDERAL TAX SUITS <input type="checkbox"/> 870 Taxes (U.S. Plaintiff or Defendant) <input type="checkbox"/> 871 IRS—Third Party 26 USC 7609	<input type="checkbox"/> 400 State Reapportionment <input type="checkbox"/> 410 Antitrust <input type="checkbox"/> 430 Banks and Banking <input type="checkbox"/> 450 Commerce <input type="checkbox"/> 460 Deportation <input type="checkbox"/> 470 Racketeer Influenced and Corrupt Organizations <input type="checkbox"/> 480 Consumer Credit <input type="checkbox"/> 490 Cable/Sat TV <input type="checkbox"/> 810 Selective Service <input type="checkbox"/> 850 Securities/Commodities/Exchange <input type="checkbox"/> 875 Customer Challenge 12 USC 3410 <input type="checkbox"/> 890 Other Statutory Actions <input type="checkbox"/> 891 Agricultural Acts <input type="checkbox"/> 892 Economic Stabilization Act <input type="checkbox"/> 893 Environmental Matters <input type="checkbox"/> 894 Energy Allocation Act <input type="checkbox"/> 895 Freedom of Information Act <input type="checkbox"/> 900 Appeal of Fee Determination Under Equal Access to Justice <input type="checkbox"/> 950 Constitutionality of State Statutes
CONTRACT	TORTS	FORFEITURE/PENALTY	BANKRUPTCY	OTHER STATUTES												
<input type="checkbox"/> 110 Insurance <input type="checkbox"/> 120 Marine <input type="checkbox"/> 130 Miller Act <input type="checkbox"/> 140 Negotiable Instrument <input type="checkbox"/> 150 Recovery of Overpayment & Enforcement of Judgment <input type="checkbox"/> 151 Medicare Act <input type="checkbox"/> 152 Recovery of Defaulted Student Loans (Excl. Veterans) <input type="checkbox"/> 153 Recovery of Overpayment of Veteran's Benefits <input type="checkbox"/> 160 Stockholders' Suits <input type="checkbox"/> 190 Other Contract <input type="checkbox"/> 195 Contract Product Liability <input type="checkbox"/> 196 Franchise	PERSONAL INJURY <input type="checkbox"/> 310 Airplane <input type="checkbox"/> 315 Airplane Product Liability <input type="checkbox"/> 320 Assault, Libel & Slander <input type="checkbox"/> 330 Federal Employers' Liability <input type="checkbox"/> 340 Marine <input type="checkbox"/> 345 Marine Product Liability <input type="checkbox"/> 350 Motor Vehicle <input type="checkbox"/> 355 Motor Vehicle Product Liability <input type="checkbox"/> 360 Other Personal Injury CIVIL RIGHTS <input type="checkbox"/> 441 Voting <input type="checkbox"/> 442 Employment <input type="checkbox"/> 443 Housing/Accommodations <input type="checkbox"/> 444 Welfare <input type="checkbox"/> 445 Amer. w/Disabilities - Employment <input type="checkbox"/> 446 Amer. w/Disabilities - Other <input type="checkbox"/> 440 Other Civil Rights	PERSONAL INJURY <input type="checkbox"/> 362 Personal Injury - Med. Malpractice <input type="checkbox"/> 365 Personal Injury - Product Liability <input type="checkbox"/> 368 Asbestos Personal Injury Product Liability PERSONAL PROPERTY <input type="checkbox"/> 370 Other Fraud <input type="checkbox"/> 371 Truth in Lending <input type="checkbox"/> 380 Other Personal Property Damage <input type="checkbox"/> 385 Property Damage Product Liability PRISONER PETITIONS <input type="checkbox"/> 510 Motions to Vacate Sentence <input type="checkbox"/> Habeas Corpus: <input type="checkbox"/> 530 General <input type="checkbox"/> 535 Death Penalty <input type="checkbox"/> 540 Mandamus & Other <input type="checkbox"/> 550 Civil Rights <input type="checkbox"/> 555 Prison Condition	<input type="checkbox"/> 610 Agriculture <input type="checkbox"/> 620 Other Food & Drug <input type="checkbox"/> 625 Drug Related Seizure of Property 21 USC 881 <input type="checkbox"/> 630 Liquor Laws <input type="checkbox"/> 640 R.R. & Truck <input type="checkbox"/> 650 Airline Regs. <input type="checkbox"/> 660 Occupational Safety/Health <input type="checkbox"/> 690 Other LABOR <input type="checkbox"/> 710 Fair Labor Standards Act <input type="checkbox"/> 720 Labor/Mgmt. Relations <input type="checkbox"/> 730 Labor/Mgmt. Reporting & Disclosure Act <input type="checkbox"/> 740 Railway Labor Act <input type="checkbox"/> 790 Other Labor Litigation <input type="checkbox"/> 791 Empl. Ret. Inc. Security Act	<input type="checkbox"/> 422 Appeal 28 USC 158 <input type="checkbox"/> 423 Withdrawal 28 USC 157 PROPERTY RIGHTS <input type="checkbox"/> 820 Copyrights <input checked="" type="checkbox"/> 830 Patent <input type="checkbox"/> 840 Trademark SOCIAL SECURITY <input type="checkbox"/> 861 HIA (1395f) <input type="checkbox"/> 862 Black Lung (923) <input type="checkbox"/> 863 DIWC/DIWW (405(g)) <input type="checkbox"/> 864 SSID Title XVI <input type="checkbox"/> 865 RSI (405(g)) FEDERAL TAX SUITS <input type="checkbox"/> 870 Taxes (U.S. Plaintiff or Defendant) <input type="checkbox"/> 871 IRS—Third Party 26 USC 7609	<input type="checkbox"/> 400 State Reapportionment <input type="checkbox"/> 410 Antitrust <input type="checkbox"/> 430 Banks and Banking <input type="checkbox"/> 450 Commerce <input type="checkbox"/> 460 Deportation <input type="checkbox"/> 470 Racketeer Influenced and Corrupt Organizations <input type="checkbox"/> 480 Consumer Credit <input type="checkbox"/> 490 Cable/Sat TV <input type="checkbox"/> 810 Selective Service <input type="checkbox"/> 850 Securities/Commodities/Exchange <input type="checkbox"/> 875 Customer Challenge 12 USC 3410 <input type="checkbox"/> 890 Other Statutory Actions <input type="checkbox"/> 891 Agricultural Acts <input type="checkbox"/> 892 Economic Stabilization Act <input type="checkbox"/> 893 Environmental Matters <input type="checkbox"/> 894 Energy Allocation Act <input type="checkbox"/> 895 Freedom of Information Act <input type="checkbox"/> 900 Appeal of Fee Determination Under Equal Access to Justice <input type="checkbox"/> 950 Constitutionality of State Statutes											
V. ORIGIN (Place an "X" in One Box Only)			<table border="1" style="width:100%; border-collapse: collapse;"> <tr> <td><input checked="" type="checkbox"/> 1 Original Proceeding</td> <td><input type="checkbox"/> 2 Removed from State Court</td> <td><input type="checkbox"/> 3 Remanded from Appellate Court</td> <td><input type="checkbox"/> 4 Reinstated or Reopened</td> <td><input type="checkbox"/> 5 Transferred from another district (specify)</td> <td><input type="checkbox"/> 6 Multidistrict Litigation</td> <td><input type="checkbox"/> 7 Appeal to District Judge from Magistrate Judgment</td> </tr> </table>			<input checked="" type="checkbox"/> 1 Original Proceeding	<input type="checkbox"/> 2 Removed from State Court	<input type="checkbox"/> 3 Remanded from Appellate Court	<input type="checkbox"/> 4 Reinstated or Reopened	<input type="checkbox"/> 5 Transferred from another district (specify)	<input type="checkbox"/> 6 Multidistrict Litigation	<input type="checkbox"/> 7 Appeal to District Judge from Magistrate Judgment				
<input checked="" type="checkbox"/> 1 Original Proceeding	<input type="checkbox"/> 2 Removed from State Court	<input type="checkbox"/> 3 Remanded from Appellate Court	<input type="checkbox"/> 4 Reinstated or Reopened	<input type="checkbox"/> 5 Transferred from another district (specify)	<input type="checkbox"/> 6 Multidistrict Litigation	<input type="checkbox"/> 7 Appeal to District Judge from Magistrate Judgment										
VI. CAUSE OF ACTION			Cite the U.S. Civil Statute under which you are filing (Do not cite jurisdictional statutes unless diversity): <u>35 U.S.C. et. seq.</u> Brief description of cause: <u>Patent infringement</u>													
VII. REQUESTED IN COMPLAINT:			<input type="checkbox"/> CHECK IF THIS IS A CLASS ACTION UNDER F.R.C.P. 23 DEMAND \$ _____ CHECK YES only if demanded in complaint: JURY DEMAND: <input checked="" type="checkbox"/> Yes <input type="checkbox"/> No													
VIII. RELATED CASE(S) IF ANY			(See instructions): JUDGE _____ DOCKET NUMBER _____													
DATE <u>5/10/06</u> SIGNATURE OF ATTORNEY OF RECORD 																
FOR OFFICE USE ONLY																
RECEIPT # _____	AMOUNT _____	APPLYING IFP _____	JUDGE _____	MAG. JUDGE _____												

AO FORM 85 RECEIPT (REV. 9/04)

United States District Court for the District of Delaware

08 - 311

Civil Action No. _____

ACKNOWLEDGMENT
OF RECEIPT FOR AO FORM 85

NOTICE OF AVAILABILITY OF A
UNITED STATES MAGISTRATE JUDGE
TO EXERCISE JURISDICTION

I HEREBY ACKNOWLEDGE RECEIPT OF 1 COPIES OF AO FORM 85.

5-10-06

(Date forms issued)

Kristen Clark

(Signature of Party or their Representative)

Kristen Clark

(Printed name of Party or their Representative)

Note: Completed receipt will be filed in the Civil Action